KLASIFIKASI TEKS BAHASA INDONESIA PADA DOKUMEN PENGADUAN SAMBAT *ONLINE* MENGGUNAKAN METODE *NAÏVE BAYES* DAN KOMBINASI SELEKSI FITUR

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Hilmy Khairi Idris
NIM: 155150201111256

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

KLASIFIKASI TEKS BAHASA INDONESIA PADA DOKUMEN PENGADUAN SAMBAT *ONLINE* MENGGUNAKAN METODE *NAÏVE BAYES* DAN KOMBINASI SELEKSI FITUR

SKRIPSI

Untuk memnuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Hilmy Khairi Idris
NIM: 155150201111256

Skripsi ini telah diuji dan dinyatakan lulus pada
27 Desember 2018
Telah diperiksa dan disetuji oleh:

Pembimbing I	Pembimbing II
Mochammad Ali Fauzi, S.Kom, M.Kom	Indriati, S.T, M.Kom
NIK: 201502 890101 1 001	NIP: 19831013 201504 2 002

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain dalam kegiatan akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripso ini terbukti terdapat unsur-unsur plagiasi, saya bersedia PKL ini digugurkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 27 Desember 2018

Hilmy Khairi Idris

NIM: 155150201111256

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa yang telah memberikan berkat sehingga skripsi yang berjudul "Klasifikasi Teks Bahasa Indonesia Pada Dokumen Pengaduan SAMBAT *Online* Menggunakan Metode *Naïve Bayes* dan Kombinasi Seleksi Fitur" ini dapat terselesaikan.

Penulis menyadari bahwa dalam penyelesaian dan penyusunan skripsi ini tidak akan terwujud tanpa adanya bantuan dan dorongan dari beberapa pihak. Oleh karena itu, dengan skripsi ini penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Mochammad Ali Fauzi, S.Kom, M.Kom selaku dosen Pembimbing I yang telah menyediakan waktu untuk membimbing penulis dalam menyelesaikan naskah skripsi ini.
2. Ibu Indriati, S.T, M.Kom selaku dosen Pembimbing II yang telah menyediakan waktu untuk membimbing penulis dalam menyelesaikan naskah skripsi ini.
3. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya Malang.
4. Bapak Agus Wahyu Widodo, ST. M.Cs. selaku Ketua Program Studi Informatika Fakultas Ilmu Komputer Universitas Brawijaya Malang.
5. Bapak Nurul Hidayat, S.Pd, M.Sc selaku dosen Penasihat Akademik yang selalu memberikan nasehat kepada penulis selama menempuh masa studi.
6. Alm. Bapak Muhammad Idris dan Ibu Wakisma selaku orang tua atas segala nasihat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti-hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
7. Nurul Fadhilah Idris dan Nurul Ariqah Idris selaku saudara yang selalu mendoakan penulis agar dapat menyelesaikan studi tepat waktu.
8. Delarta Tok Adin, Septian Dwi Cahyo, Prita Nur Rizky Faridiani, Katherine Ivana Ruslim, Fera Fanesya, Salma Hanifah, dan Meidita Famirah selaku teman-teman kuliah yang selalu membantu, mendorong, dan membagi ilmunya selama perkuliahan.
9. Seluruh keluarga besar Dara Daeng Brawijaya (Komunitas Mahasiswa Sulawesi Selatan di Universitas Brawijaya) yang selalu menjadi penyemangat dalam menyelesaikan naskah skripsi ini.
10. Teman-teman mahasiswa Informatika 2015 Universitas Brawijaya atas segala dukungan dan dorongan selama perkuliahan.
11. Pihak-pihak lain yang secara langsung maupun tidak langsung dalam penyelesaian naskah skripsi ini.

Penulis menyadari bahwa dalam masa penyusunan skripsi ini masih banyak kekurangan, dengan itu penulis sangat mengharapkan saran dan kritik yang membangun untuk skripsi ini. Semoga skripsi ini dapat membawa manfaat bagi semua pihak dan digunakan sebagaimana mestinya.

Malang, 27 Desember 2018

Penulis
Email: hilmykhairii@student.ub.ac.id

ABSTRAK

Hilmy Khairi Idris, Klasifikasi Teks Bahasa Indonesia Pada Dokumen Pengaduan SAMBAT *Online* Menggunakan Metode *Naïve Bayes* dan Kombinasi Seleksi Fitur

Pembimbing: Mochammad Ali Fauzi, S.Kom, M.Kom dan Indriati, S.T, M.Kom

SAMBAT *Online* merupakan bentuk realisasi *e-Government* yang ada di Kota Malang. SAMBAT *Online* atau Sistem Aplikasi Masyarakat Bertanya Terpadu *Online* suatu platform berupa situs web yang disediakan oleh pemerintah Kota Malang untuk menerima laporan pengaduan, kritik, saran, maupun pertanyaan kepada pemerintah. Setiap laporan yang masuk akan dikelompokkan secara manual oleh pengelola sistem SAMBAT *Online*. Pengelompokan yang dilakukan berdasarkan Satuan Kerja Perangkat Daerah (SKPD) yang dituju secara manual. Oleh karena ini dibangunlah sistem klasifikasi untuk mengefesiensikan waktu pada proses pengelompokan laporan ke SKPD yang dituju menggunakan metode *Naïve Bayes* dan Kombinasi Seleksi Fitur antara *Chi-Square* dan *Information Gain.* Pada pengujian yang dilakukan, sistem berhasil memberikan hasil akurasi yang lebih baik apabila menggunakan selsksi fitur dibanding tanpa menggunakan seleksi fitur dengan nilai akurasi sebesar 83,33%. Selanjutnya pada saat dilakukan kombinasi seleksi fitur, hasil akurasi yang didapatkan sama dengan hasil tanpa dilakukan kombinasi yaitu 83,33%. Sehingga kombinasi seleksi fitur belum bisa memberikan hasil yang lebih baik.

Kata kunci: *SAMBAT Online, Klasifikasi Teks, Chi-Square, Information Gain, Naïve Bayes*

ABSTRACT

Hilmy Khairi Idris, ***Indonesian Text Classification on SAMBAT Online Complaint Documents Using the Naïve Bayes Method and Feature Selection Combination***

***Supervisors*: Mochammad Ali Fauzi, S.Kom, M.Kom *and* Indriati, S.T, M.Kom**

SAMBAT Online is a form of e-Government realization in Malang City. SAMBAT Online or The Integrated Online Community Application System is a platform of a website provided by the Malang City Government to receive complaints, criticisms, suggestions, or questions to the government. Each incoming report will be grouped manually by the SAMBAT Online system manager. Grouping is based on The Regional Work Unit (SKPD) which is handled manually. Therefore, a classification system was built to save time in the process of grouping reports to SKPD using the Naïve Bayes method and the Combination of Feature Selection between Chi-Square and Information Gain. In the tests conducted, the system succeeded in providing better accuracy results when using feature selection than without using feature selection with an accuracy value of 83.33%. Furthermore, when a feature selection combination is performed, the results of the accuracy obtained are the same as the results without a combination of 83.33%. So, the combination of selection has not been able to provide better results.

Keywords: *SAMBAT Online, Text Classification, Chi-Square, Information Gain, Naïve Bayes*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Pelayanan yang dilakukan oleh institusi pemerintah secara optimal, efektif dan sesuai dengan standar merupakan bagian dari standar setiap intitusi di Indonesia (Somantri, *et al*., 2017). Untuk mewujudkan pelayanan yang optimal memerlukan sebuah sistem yang baik dan terintegrasi mulai dari tingkat pusat sampai pada tingkat daerah. *E-Government* merupakan sebuah sistem terintegrasi dengan menggunakan media teknologi informasi dalam pelaksanaannya. *E-Government* atau *eGov* merupakan sebuah proses pemanfaatan teknologi informasi sebagai alat bantu pemerintah dalam menjalankan sistem pemerintahan secara efisien yang dapat meningkatkan hubungan pemerintah dengan pihak lain (Somantri, *et al*., 2017).

Sebanding dengan terus berkembangnya Kota Malang, pemerintah Kota Malang terus meningkatkan kualitas pelayanannya terhadap masyarakat. Salah satunya dengan penerapan *e-Government*. SAMBAT *Online* merupakan bentuk realisasi *e-Government* yang ada di Kota Malang. SAMBAT *Online* atau Sistem Aplikasi Masyarakat Bertanya Terpadu *Online* suatu platform berupa situs web yang disediakan oleh pemerintah Kota Malang untuk menerima laporan pengaduan, kritik, saran, maupun pertanyaan kepada pemerintah. Masyarakat Kota Malang dapat memberikan laporannya melalui pesan singkat atau mengirimkan langsung melalui situs web yang sudah disediakan.

Setiap laporan yang masuk akan dikelompokkan secara manual oleh pengelola sistem SAMBAT *Online*. Pengelompokan yang dilakukan berdasarkan Satuan Kerja Perangkat Daerah (SKPD) yang dituju. Pengelola sistem berhak tidak merespon ataupun menampilkan laporan yang masuk apabila tidak sesuai prosedur yang ada karena setiap laporan yang masuk akan diperiksa terlebih dahulu apakah sesuai dengan prosedur pelaporan SAMBAT *Online*. Klasifikasi teks dapat dilakukan untuk mengefisiensikan waktu pada proses pengelompokan laporan ke SKPD yang dituju oleh pengelola sistem SAMBAT *Online*.

Peneletian sebelumnya dalam melakukan klasifikasi pengaduan SAMBAT *online* menggunakan metode *KNN* menghasilkan akurasi sebesar 78% (Suharno, *et al.,* 2017). Selain itu dengan objek yang sama dengan menggunakan metode *N-Gram* dan *NW-KNN* menghasilkan akurasi sebesar 75% (Prasanti, *et al.,* 2017). Selain metode tersebut masih banyak metode klasifikasi yang dapat digunakan salah satunya adalah *Naïve Bayes*. Metode *Naïve Bayes* merupakan metode klasifikasi sederhana tetapi memiliki akurasi dan performansi yang tinggi dalam pengklasifikasian teks (Nugroho, *et al*., 2016). Pada penelitian yang dilakukan oleh Didik Garbian Nugroho yang menggunakan metode *Naïve Bayes* dalam melakukan pengklasifikasian analisis sentimen pada jasa ojek *online* menghasilkan akurasi sebesar 80%. Oleh karena itu, metode *Naïve Bayes* diharapkan dapat

menghasilkan akurasi lebih baik dari penelitian sebelumnya dalam melakukan klasifikasi teks Bahasa Indonesia pada pengaduan SAMBAT *Online*.

Dalam melakukan pengklasifikasian teks, setiap dokumen dapat menjadi milik banyak kategori. Untuk itu dilakukan tahapan seleksi fitur yang dapat meningkatkan skalabilitas, efisiensi, dan akurasinya (Zheng, *et al*., 2014). Seleksi fitur dapat dilakukan dengan banyak metode sebagai contoh nya *Mutual Information*, *Information Gain*, *Chi-Square Statistic*, dan lain-lain.

Pada penelitian ini menggunakan kombinasi seleksi fitur. Seleksi fitur yang digunakan adalah terdiri dari dua seleksi fitur. Pertama adalah *Information Gain* dan kedua adalah *Chi-Square. Information Gain* merupakan algoritma seleksi fitur yang efisien dalam mengukur jumlah bit informasi yang diperoleh pada proses klasifikasi untuk mengetahui keberadaan suatu fitur pada sebuah dokumen kemudian memilih subset optimal (Putra, *et al*., 2016). Pada penelitian Ida Bagus Gede Widnyana Putra yang menggunakan seleksi fitur *Information Gain*, memberikan hasil akurasi diatas 90%. Seleksi fitur kedua adalah *Chi-Square* yang merupakan salah satu metode seleksi fitur yang dapat membuang banyak jumlah fitur tanpa mempengaruhi tingkat akurasinya (Sun, *et al*., 2009). Pada penelitian yang dilakukan Claudio Fresta Suharno yang menggunakan seleksi fitur *Chi-Square* memberikan hasil *F-Measure* yang cenderung semakin menurun seiring dengan bertambahnya rasio jumlah fitur yang digunakan.

Dasar pemikiran kombinasi seleksi fitur ini berasal dari bidang *ensemble learning*, dimana kombinasi penggolongan untuk mendapatkan hasil yang lebih stabil, dan hasilnya kombinasi penggolongan memiliki kinerja lebih baik. Oleh karena itu, orang bisa berpikir juga pada teknik seleksi fitur dengan menggabungkan dapat menghasilkan lebih stabil dibanding tunggal. (Saeys, *et al*., 2008).

Perpaduan metode *Naïve Bayes* dengan kombinasi seleksi fitur, diharapkan dapat mengetahui hasil dari sistem klasifikasi teks dengan metode *Naïve Bayes* yang dipadukan dengan metode untuk menunjang sistem klasifikasi teks yaitu metode kombinasi seleksi fitur. Selain itu dapat digunakan sebagai acuan pada implementasi pada klasifikasi dokumen laporan SAMBAT *Online* dengan otomatis serta dapat menghasilkan tingkat akurasi yang lebih baik dari penelitian sebelumnya.

1.2 Rumusan masalah

Berdasarkan latar belakang masalah yang telah diuraikan di atas, maka didapatkan rumusan masalah sebagai berikut:

1. Bagaimana pengaruh seleksi fitur terhadap klasifikasi teks Bahasa Indonesia menggunakan metode *Naïve Bayes* pada dokumen pengaduan SAMBAT *Online*?
2. Bagaimana pengaruh kombinasi terhadap seleksi fitur dalam pengklasifikasian teks Bahasa Indonesia menggunakan metode *Naïve Bayes* pada dokumen pengaduan SAMBAT *Online*?

1.3 Tujuan

Berdasarkan permasalahan yang diuraikan sebelumnya, maka tujuan dari penelitian kali ini dapat dilihat sebagai berikut:

1. Mengetahui pengaruh seleksi fitur terhadap pengklasifikasian teks bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* menggunakan metode *Naïve Bayes.*
2. Mengetahui pengaruh kombinasi terhadap seleksi fitur dalam pengklasifikasian teks Bahasa Indonesia menggunakan metode *Naïve Bayes* pada dokumen pengaduan SAMBAT *Online.*

1.4 Manfaat

Manfaat yang didapatkan dari penilitian ini adalah:

1. Memberikan kemudahan kepada pengelola sistem pengaduan SAMBAT *Online* dalam proses mengklasifikasian dokumen pengaduan dari masyarakat Malang sesuai dengan SKPD (Satuan Kerja Perangkat Daerah) yang dituju.
2. Dapat memberikan hasil yang optimal dalam proses klasifikasi karena waktu yang dibutuhkan menjadi lebih sedikit untuk mengetahui SKPD (Satuan Kerja Perangkat Daerah) yang dituju dibandingkan dengan cara manual.

1.5 Batasan masalah

Untuk merumuskan permasalahan yang lebih terfokus dan tidak meluas maka dibuat batasan-batasan yang ditentukan pada penelitian ini yaitu:

1. Data set penelitian ini didapatkan dari dokumen pengaduan SAMBAT *Online* yang masuk ke pemerintah Kota Malang.
2. Data SKPD (Satuan Kerja Perangkat Daerah) yang digunakan sebanyak 3 SKPD.

1.6 Sistematika pembahasan

Bagian ini berisi struktur proposal tugas akhir mulai Bab Pendahuluan sampai Bab Penutup dengan deskripsi singkat dari deskripsi singkat dari masing-masing bab. Susunan dari Pembahasan sebagai berikut:

BAB 1 PENDAHULUAN

Bab ini menjelaskan tentang latar belakang dilakukannya penilitian pada klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* menggunakan metode *Naïve Bayes* dan kombinasi seleksi fitur, rumusan masalah, batasan masalah, tujuan, manfaat, serta sistematika dari penulisan.

BAB 2 TINJAUAN PUSTAKA

Bab ini menjelaskan dasar teori dan referensi yang dibutuhkan dalam pemahaman permasalahan yang dibahas dalam pembuatan tugas akhir. Teori-teori yang terdapat dalam bab ini mencakup data penelitian, SAMBAT *Online*,

klasifikasi teks, seleksi fitur, kombinasi seleksi fitur, *Simmilarity Measure*, dan *Naïve Bayes*.

BAB 3 METODOLOGI

Bab ini menjelaskan langkah-langkah yang akan dilakukan dalam penelitian yang meliputi studi literatur klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* menggunakan metode *Naïve Bayes* dan kombinasi seleksi fitur, pengumpulan data, analisa kebutuhan sistem, perancangan sistem, pengujian dan evaluasi sistem.

BAB 4 PERANCANGAN

Bab ini membahas mengenai proses perancangan yang dilakukan dalam penelitian.

BAB 5 IMPLEMENTASI

Bab ini menyajikan implementasi metode pada program untuk menentukan hasil klasifikasi dokumen SAMBAT *Online*.

BAB 6 PENGUJIAN DAN ANALISIS

Bab ini menyajikan hasil pengujian dan menganalisis hasil pengujian, yang dilakukan pada beberapa data.

BAB 7 PENUTUP

Bab ini berisi kesimpulan dan saran dari seluruh penelitian, sehingga dapat digunakan untuk pengembangan penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini berisi kajian pustaka dan pembahasan tentang teori dasar yang berhubungan dengan *preprocessing* dan klasifikasi teks menggunakan metode *Naïve Bayes* dan kombinasi seleksi fitur dalam menentukan klasifikasi pengaduan berdasarkan tugas dan fungsi SKPD. Kajian pustaka membahas penelitian yang telah ada dan yang akan diusulkan. Dasar teori membahas teori yang diperlukan untuk menyusun penelitian yang diusulkan meliputi konsep dasar dari SAMBAT *Online*, pengertian *text mining*, *text preprocessing*, kombinasi seleksi fitur, dan *Naïve Bayes*.

2.1 Kajian Pustaka

Kajian pustaka pada penelitian ini akan membahas penelitian sebelumnya yang dapat mendukung pengerjaan penelitian ini. Terdapat penelitian yang akan dibahas pada kajian pustaka ini, yaitu:

1. Penelitian yang dilakukan oleh Claudio Fresta Suharno, M. Ali Fauzi, Rizal Setya Perdana (2017) yang membahas klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* menggunakan metode *K-Nearest Neighbor* dan *Chi-Square*.
2. Penelitian yang dilakukan oleh Didik Garbian Nugroho, Yulison Herry Chrisnanto, Agung Wahana (2016) yang membahas perihal Analisis Sentimen Pada Jasa Ojek Online Menggunakan Metode *Naïve Bayes.*
3. Penelitian yang dilakukan oleh Annisya Aprilia Prasanti, M. Ali Fauzi, dan M. Tanzil Furqon (2017) yang membahas klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* menggunakan metode *N-Gram* dan *Neighbor Weighted K-Nearest Neighbor (NW-KNN)*.
4. Penelitian yang dilakukan oleh Firda Ika Pratiwi, Rekyan Regasari Mardi, dan Budi Darma Setiawan (2015) yang membahas klasifikasi topik pada skripsi berdasarkan judul dan abstraksi dengan menggunakan metode *Transformed Weight-Normalized Complement Naïve Bayes*.
5. Penelitian yang dilakukan oleh Ida Bagus Gede Widnyana Putra, Made Sudarma, dan I Nyoman Satya Kumara (2016) yang membahas klasifikasi teks bahasa Bali dengan metode *supervised learning Naive Bayes Classifier.*

Pada penelitian yang dilakukan oleh Claudio Fresta Suharno, M. Ali Fauzi, Rizal Setya Perdana (2017), bertujuan untuk melakukan klasifikasi dokumen pengaduan SAMBAT *online* dengan menggunakan metode *K-Nearest Neighbor* dan metode seleksi fitur *Chi-Square*. Dari penelitian ini menunjukkan bahwa hasil yang didapatkan dengan menggunakan seleksi fitur lebih baik daripada tanpa adanya proses seleksi fitur. *Precision* dan *recall* terbaik didapatkan pada k = 15 dengan seleksi fitur sebesar 25%. Sedangkan hasil dari *F-Measure* terbaik didapatkan dengan nilai 78% pada k = 15 dan k = 5 dengan seleksi fitur sebesar 25%.

Pada penelitian yang dilakukan oleh Didik Garbian Nugroho, Yulison Herry Chrisnanto, dan Agung Wahana (2016), bertujuan untuk melakukan analisis sentimen pada jasa ojek *online* menggunakan metode *Naïve Bayes*. Implementasi algoritme ini mampu mengklasifikasi sentimen menggunakan Naïve Bayes dengan akurasi yang dihasilkan sebesar 80% berdasarkan 800 data tweet yang terdiri atas 300 data latih dan 500 data uji. Terdapat kesalahan pada data uji fitur yang muncul tidak sesuai dengan klasifikasinya. Akurasi klasifikasi dapat ditingkatkan dengan menambah jumlah data latih.

Pada penelitian yang dilakukan oleh Annisya Aprilia Prasanti, M. Ali Fauzi, dan M. Tanzil Furqon (2017) yang bertujuan untuk melakukan klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* menggunakan metode *N-Gram* dan *Neighbor Weighted K-Nearest Neighbor (NW-KNN)*. Berdasarkan hasil pengujian penelitian ini, dapat disimpulkan bahwa algoritme NW-KNN mampu melakukan klasifikasi teks pengaduan dengan nilai tetangga k terdekat yang paling optimal adalah 3, dengan rata-rata hasil presentase *precision* sebesar 77.85%, rata-rata hasil presentase *recall* sebesar 74.18% dan rata-rata hasil presentase *f-measure* sebesar 75.25%.

Pada penelitian yang dilakukan oleh Firda Ika Pratiwi, Rekyan Regasari Mardi, dan Budi Darma Setiawan (2015) yang bertujuan untuk melakukan klasifikasi topik pada skripsi berdasarkan judul dan abstraksi dengan menggunakan metode *Transformed Weight-Normalized Complement Naïve Bayes*. Hasil yang telah diperoleh melalui tahap pengujian adalah jumlah data latih dan jumlah fitur kata berpengaruh terhadap hasil akurasi. Rata-rata nilai *f-measure* terbaik yaitu pada pengujian pengaruh jumlah fitur kata sebesar 57% sehingga dapat disimpulkan bahwa sistem sudah berjalan relatif cukup baik.

Penelitian yang dilakukan oleh Ida Bagus Gede Widnyana Putra, Made Sudarma, dan I Nyoman Satya Kumara (2016) yang membahas klasifikasi teks bahasa Bali dengan metode *supervised learning Naive Bayes Classifier.* Berdasarkan hasil pengujian penelitian ini, *recall* dan akurasi antara setiap *fold* memiliki nilai yang berdekatan dengan nilai rata-rata *precision* 97,587 %, *recall* 92,778 % dan akurasi sebesar 95,222 %.

Sistem pengklasifikasian yang akan dibangun adalah berupa sistem pengklasifikasian yang menggunakan metode *Naive Bayes* dan kombinasi seleksi fitur. Metode ini diharapkan dapat mengklasifikasikan dokumen ke SKPD tertentu seperti halnya dilakukan admin SAMBAT *Online* secara manual. Perbandingan objek dan hasil penelitian sebelumnya di tunjukkan pada Tabel 2.1

Tabel 2.1 Perbandingan Objek, Metode dan Hasil Penilitian Sebelumnya

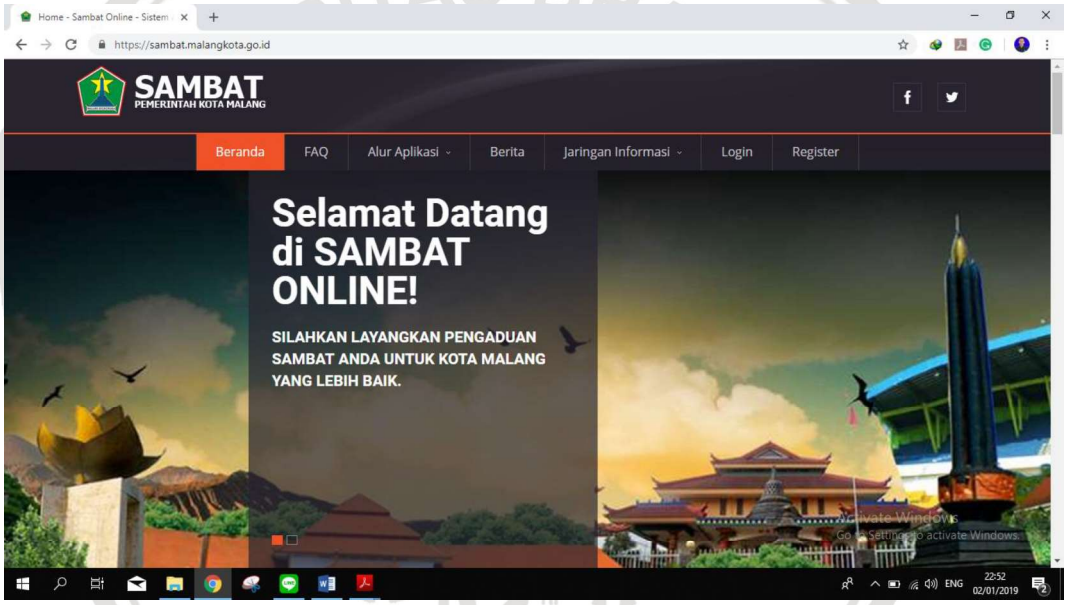
No.	Nama Penulis	Judul	Tahun	Keterangan
1	Claudio Fresta Suharno, M. Ali	Klasifikasi Teks Bahasa Indonesia Pada	2017	*Precision* dan *recall* terbaik didapatkan pada k = 15 dengan seleksi fitur

	Fauzi, Rizal Setya Perdana	Dokumen Pengaduan SAMBAT *Online* Menggunakan Metode K-Nearest Neighbor Dan *Chi-Square*.		sebesar 25%. Sedangkan hasil dari F-Measure terbaik didapatkan dengan nilai 78% pada k = 15 dan k = 5 dengan seleksi fitur sebesar 25%.
2	Didik Garbian Nugroho, Yulison Herry Chrisnanto, Agung Wahana	Analisis Sentimen Pada Jasa Ojek Online Menggunakan Metode *Naïve Bayes*	2016	Sistem mampu mengklasifikasi sentimen menggunakan Naïve Bayes dengan akurasi yang dihasilkan sebesar 80% berdasarkan 800 data tweet yang terdiri atas 300 data latih dan 500 data uji.
3	Annisya Aprilia Prasanti, M. Ali Fauzi, dan M. Tanzil Furqon	Klasifikasi Teks Bahasa Indonesia Pada Dokumen Pengaduan SAMBAT *Online* Menggunakan Metode *N-Gram* dan *Neighbor Weighted K-Nearest Neighbor (NW-KNN).*	2017	Klasifikasi teks pengaduan dengan nilai tetangga k terdekat yang paling optimal adalah 3, dengan rata-rata hasil presentase *percision* sebesar 77.85%, rata-rata hasil presentase *recall* sebesar 74.18% dan rata-rata hasil presentase *f-measure* sebesar 75.25%.
4	Firda Ika Pratiwi, Rekyan Regasari Mardi, dan Budi Darma Setiawan	Klasifikasi Topik Pada Skripsi Berdasarkan Judul Dan Abstraksi Dengan Menggunakan Metode *Transformed Weight-Normalized Complement Naïve Bayes.*	2015	Jumlah data latih dan jumlah fitur kata berpengaruh terhadap hasil akurasi. Rata-rata nilai *f-measure* terbaik yaitu pada pengujian pengaruh jumlah fitur kata sebesar 57% sehingga dapat disimpulkan bahwa sistem sudah berjalan relative cukup baik.
5	Ida Bagus Gede Widnyana	Klasifikasi Teks Bahasa Bali	2016	*Recall* dan akurasi antara setiap *fold* memiliki nilai

	Putra, Made Sudarma, dan I Nyoman Satya Kumara	Dengan Metode *Upervised Learning Naive Bayes Classifier.*		yang berdekatan dengan nilai rata-rata *precision* 97,587 %, *recall* 92,778 % dan akurasi sebesar 95,222 %.

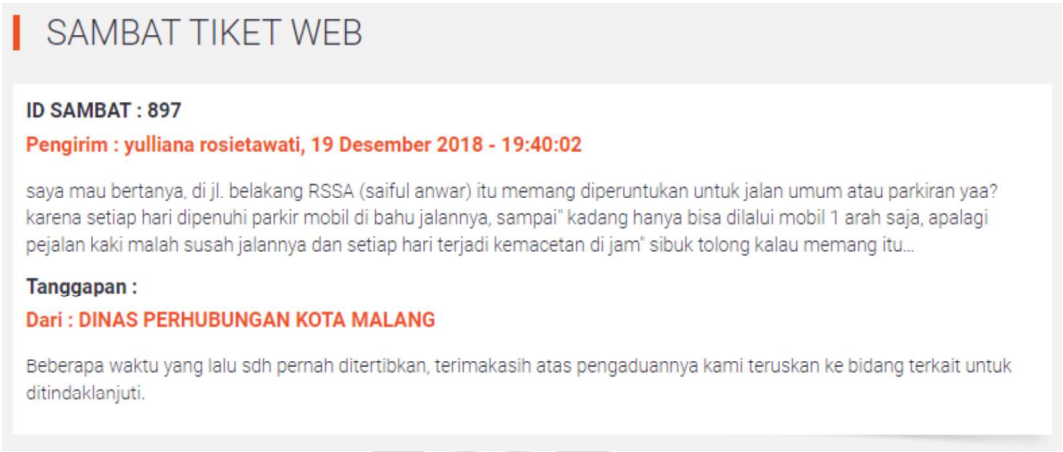
2.2 SAMBAT *Online*

SAMBAT *Online* atau Sistem Aplikasi Masyarakat Bertanya Terpadu *Online* merupakan fasilitas yang disediakan oleh Dinas Komunikasi dan Informatika (Diskominfo) untuk masyarakat kota Malang yang ingin menyampaikan pengaduan, kritik, saran, maupun pertanyaan terhadap permasalahan yang ada di wilayah kota Malang terkait apapun yang ingin disampaikan atau ditanyakan ke pemerintah kota Malang. Berikut tampilan SAMBAT *online* yang dapat dilihat pada Gambar 2.1.



Gambar 2.1 Tampilan SAMBAT *Online*

Dalam melakukan pelaporan, masyarakat kota Malang dapat mengirimkan laporannya melalui dua jalur, yaitu melalui website secara langsung maupun melalui pesan singkat. Laporan yang masuk akan diklasifikasikan secara manual untuk diteruskan ke SKPD yang terkait. Contoh pelaporan yang masuk pada SAMBAT *Online* dapat dilihat pada Gambar 2.2.



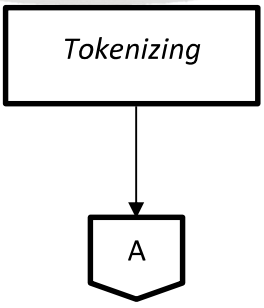
Gambar 2.2 Contoh Pelaporan SAMBAT *Online*

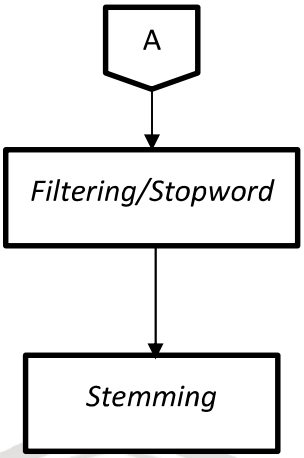
2.3 Text Mining

Text Mining adalah sebuah sistem dimana pengguna dapat berinteraksi dengan berbagai macam dokumen. Dalam *text mining* kita dapat mengekstraksi berbagai macam informasi pada sebuah dokumen (Indriati, *et al*., 2016). Tujuan *text mining* untuk menganalisis hubungan antar dokumen dengan mengambil kata yang dapat mewakili dokumen tersebut. Oleh karena itu, *text mining* memiliki peran penting dalam bidang *data mining*. Dengan mengaplikasikan proses-proses dalam *text mining*, maka akan diperoleh pola-pola data, tren, dan ekstraksi dari pengetahuan-pengetahuan yang potensial dari data teks (Hidayatullah, *et al*., 2016).

2.3.1 Text Preprocessing

Text preprocessing adalah proses awal yang dilakukan dalam melakukan klasifikasi teks untuk mendapatkan dokumen yang siap untuk diolah. *Tahapan text preprocessing* setelah membaca teks input dokumen, proses ini membagi dokumen teks ke fitur yang disebut (*tokenization*, kata, istilah atau atribut), setelah itu menghilangkan yang tidak informatif (Khadim, *et al*., 2014). Dari mulai inputan teks tersebut kemudian melalui banyak tahapan untuk menghasilkan suatu dokumen latih dan dokumen uji yang masih berbentuk mentah dan siap untuk diolah. *Preprocessing* juga sangat berguna untuk mengubah dokumen inputan awal yang diterima menjadi dokumen yang lebih rapi. Untuk proses *preprocessing* dapat dilakukan dengan banyak cara yang pada umumnya dapat dilihat pada Gambar 2.3.





Gambar 2.3. Diagram Alir *Preprocessing*

2.3.1.1 *Tokenizing*

Tokenizing merupakan proses yang bertujuan untuk mengambil kata dari penyusun suatu dokumen. *Tokenizing* merupakan tahap dimana yang harus dilalui sebelum melakukan tahapan selanjutnya dalam *text mining*. Proses ini mengubah semua huruf kapital ke huruf kecil. Menghilangkan karakter-karakter yang tidak berpengaruh pada *text preprocessing* juga dilakukan. Karakter-karakter tersebut dapat berubah spasi, tanda baca, angka serta karakter lain selain huruf (Prasanti, *et al*., 2017). *String* akan terlihat lebih rapi untuk diolah karena hasil dari *tokenizing* menampilkan kata per kata. Tiap kata dan istilah tersebut berupa kata tunggal yang menyusun suatu dokumen. Pada tahap ini, dilakukan pemotongan (*parsing*) terhadap kata tunggal tersebut menjadi kumpulan token.

2.3.1.2 *Filtering/Stopwords*

Pada tahap ini dilaksanakan proses filter atau penyaringan kata hasil dari proses Tokenizing, dimana kata yang tidak relevan terhadapat dokumen dan yang sering muncul dalam dokumen akan dibuang (Pratiwi, *et al*., 2015). Proses ini dilakukan dengan melibatkan daftar kata yang tidak berpengaruh banyak terhadap dokumen atau sering disebut dengan *stoplist*. Kata-kata yang terdapat pada dokumen dibandingkan dengan stop*list* untuk menghilangkan kata-kata yang terdapat pada stop*list*. Contoh *stoplist* dalam Bahasa Indonesia adalah “di”, “ke”, “yang”, dan sebagainya. Salah satu *stoplist* Bahasa Indonesia yang dapat digunakan adalah *Stoplist* Tala. Untuk contoh penerapannya pada sebuah dokumen bisa dilihat sebagai berikut:

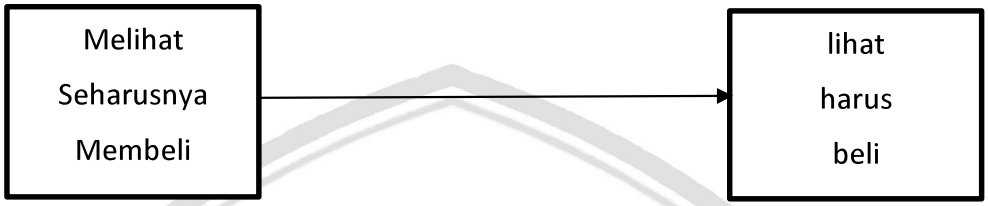
Input : Mahasiswa Fakultas Ilmu Komputer berkunjung ke kantor Google.

Output : Mahasiswa Fakultas Ilmu Komputer berkunjung kantor Google.

2.3.1.3 *Stemming*

Stemming adalah proses penghapusan afiks (prefiks dan sufiks) dari fitur-fitur yaitu proses yang diturunkan untuk mengurangi kata-kata yang terinfleksi (atau

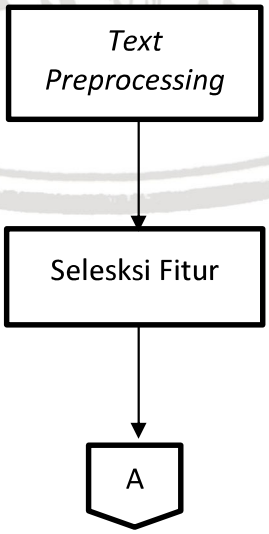
kadang-kadang diturunkan) ke kata dasar (Khadim, *et al*., 2014). Stem adalah bentuk dasar dari sebuah kata setelah menghilangkan imbuhan awal maupun imbuhan akhirnya. Proses ini digunakan untuk mengurangi jenis kata berimbuhan yang memiliki kata dasar yang sama agar tidak terlalu banyak jenis kata yang digunakan saat proses klasifikasi. Salah satu algoritme *stemmer* yang dapat digunakan adalah *stemmer* Sastrawi. Untuk contoh dari proses stemming dapat dilihat pada Gambar 2.4.

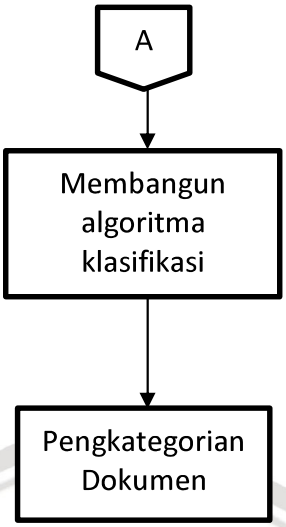


Gambar 2.4 Contoh Proses *Stemming*

2.4 Klasifikasi Teks

Klasifikasi merupakan tahap menemukan suatu model yang menggambarkan atau membedakan kelas data yang bertujuan untuk memprediksi kelas dari suatu objek baru (Prasanti, *et al*., 2017). Klasifikasi teks adalah salah satu pengaplikasian Text Mining dalam pengelompokan data. Penelitian tentang klasifikasi teks selalu menjadi topik hangat di bidang penambangan teks dibeberapa tahun terakhir (Li, *et al*., 2016). Metode klasifikasi sangat banyak yang dapat digunakan yaitu *Decision/class classification trees*, *Bayesian classifiers/Naïve Bayes classifiers*, *Neural Network*, Analisis Statistik, Algoritme Genetika, *Rough sets*, *Memory based reasoning*, dan *Support Vector Machine (SVM)*. Pada penelitian kali ini metode klasifikasi yang digunakan adalah Naïve Bayes Classifier. Untuk tahapan klasifikasi teks pada umunya dapat dilihat pada Gambar 2.5.





Gambar 2.5 Proses Klasifikasi Teks

2.5 Seleksi Fitur

Pemilihan fitur telah diterapkan untuk kategorisasi teks untuk meningkatkan skalabilitas, efisiensi, dan akurasi (Zheng, *et al*., 2014). Seleksi fitur adalah salah satu bagian dari tahapan *preprocessing*. Pada tahapan *preprocessing* seperti *stemming* dan *filtering/stopwords* masih terbilang kurang sehingga seleksi fitur dilakukan. Seleksi fitur untuk menentukan fitur yang penting untuk digunakan.

Tujuan dari seleksi fitur adalah untuk meningkatkan performa klasifikasi teks dengan menghilangkan fitur yang dianggap tidak relevan dalam klasifikasi untuk mengurangi dimensi dari himpunan (Suharno, *et al*., 2017). Sejumlah metrik seleksi fitur telah dieksplorasi, beberapa yang terkemuka di antaranya adalah *Information Gain* (IG), *Chi-square* (CHI), *Coefficient Correlation* (CC), dan *Odds Ratio* (OR). Untuk penelitian ini menggunakan seleksi fitur dari *Chi-square* dan *Information Gain* yang dikombinasikan.

2.5.1 Information Gain

Pada penelitian kali ini menggunakan dua seleksi fitur. Seleksi fitur yang pertama adalah *Information Gain* (IG). *Information Gain* merupakan seleksi fitur dimana ukuran jumlah bit informasi yang diperoleh untuk prediksi kategori dengan mengetahui ada atau tidaknya istilah dalam dokumen (Zheng, *et al*., 2016). Persamaan *Information Gain* dapat dilihat sebagai berikut:

$$IG(w_i, d) = P(w_i) \sum P(d_k|w_i) \log \mathrm{P}(d_k|w_i) + P(\overline{w_i}) P(d_k|\overline{w_i}) \log \sum P(d|\overline{w_i}) \qquad (2.1)$$

Pada persamaan di atas, $P(w_i)$ adalah probabilitas kata w_i muncul, berarti kata w_i tidak terjadi, kemudian $P(d_k)$ adalah probabilitas nilai dokumen kth, $P(d_k|w_{i,j})$ adalah probabilitas kondisional dari nilai dokumen k yang diberikan w_i,

$P(w_{i,j})$ adalah probabilitas w_i dan w_j muncul bersama, dan i, j berarti w_i dan w_j tidak muncul bersama tetapi w_i atau w_j dapat muncul (Fauzi, *et al*., 2017).

2.5.2 Chi-Square

Seleksi fitur yang kedua adalah *Chi-Square*. *Chi-Square* merupakan metode statistika pengujian hipotesis data diskrit yang menentukan apakah sebuah variabel tersebut saling berkaitan atau tidak dan mengevaluasi antara dua variable seberapa besar korelasinya. Persamaan *Chi-Square* sebagai berikut (Suharno, *et al*., 2017).

$$X^2(t,c) = \frac{N(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)} \quad (2.2)$$

Pada persamaan di atas, t adalah kata dan c adalah kelas atau kategori. Kemudian N adalah jumlah dokumen latih. A merupakan jumlah banyaknya dokumen pada kategori c yang memuat t, B jumlah banyaknya dokumen bukan kategori c yang memuat t, C jumlah banyaknya dokumen pada kategori c yang tidak memuat t, dan D adalah jumlah banyaknya dokumen bukan kategori c yang tidak memuat t.

2.6 Naïve Bayes

Metode *Naïve Bayes* melalui tahap *training* dan klasifikasi pada proses klasifikasinya. Pada tahap pelatihan dilakukan proses analisis pada sampel dokumen berupa pemilihan *vocabulary*, yaitu kata yang mungkin muncul dalam koleksi dokumen sampel yang sedapat mungkin dapat menjadi representasi dokumen (Hamzah, 2012). Kemudian dari sampel dokumen dicari probabilitas prior nya pada setiap kategori. Untuk tahapan klasifikasinya dari satu dokumen ditentukan nilai kategorinya berdasarkan term yang muncul dalam dokumen yang diklasifikasi. Persamaan *Naïve Bayes* bisa dilihat sebegai berikut:

$$(C_j|W_i) = \frac{P(C_j)\times P(W_i|C_j)}{P(W_i)} \quad (2.3)$$

Keterangan :

$P(C_j|W_i)$: Perhitungan *Posterior*, dimana peluang kemunculan kategori j dengan syarat kemunculan kata i.

$P(C_j)$: Perhitungan *Prior*, dimana peluang kemunculan setiap dokumen pada kategori j.

$P(W_i|C_j)$: Perhitungan *Likelihood* atau *Conditional Probability*, dimana peluang setiap kata i dengan syarat kategori j.

$P(W_i)$: Perhitungan *Evidence,* dimana peluang kemunculan setiap kata.

i : indeks untuk menyimpan kata dari kata 1 sampai dengan kata ke-n.

j : indeks untuk menyimpan kategori dari kategori 1 sampai dengan kategori ke-n.

Dalam pembuatan sistem ini, metode yang diterapkan adalah *Multinomial Naive Bayes*. *Multinomial Naive Bayes* merupakan pengembangan dari *Naive*

Bayes dimana kelas dokumen ditentukan melalui jumlah kemunculan kata yang ada di dalam dokumen tanpa memperhitungkan urutan kata tersebut (Destuardi dan Surya, 2009). Adapun dalam perhitungannya menggunakan Persamaan 2.4.

$$P(W_i|C_j) = \frac{count(W_i,C_j)+1}{(\sum_{w\in V} count(W_i,C_j))+|V|} \quad (2.4)$$

Keterangan :

$P(W_i|C_j)$: Peluang munculnya kata i di dalam kategori j.

$count(W_i, C_j)$: jumlah kata i pada kategori j.

$\sum_{w\in V} count(W_i, C_j)$: jumlah seluruh kata yang terdapat pada kategori j.

$|V|$: jumlah kata unik yang terdapat dalam semua kategori.

2.7 Evaluasi

Evaluasi adalah proses yang dilakukan untuk melihat keberhasilan suatu sistem dengan cara membandingakan antara kriteria dengan standar tertentu dengan hasil implementasi. Dari hasil evaluasi maka akan mendapat informasi dari keberhasilan suatu kegiatan sehingga diketahui bila terdapat selisih antara standar uang ditetapkan dengan hasil yang telah dicapai (Prasanti, *et al*., 2017).

Untuk mengukur tingkat keberhasilan dapat dilakukan untuk mengukur validitas hasil klasifikasi dengan menghitung nilai *precision, recall*, dan *f-meassure*. Perhitungan nilai *precision* akan mengukur tingkat kepastian (*exactness*) atau jumlah data *testing* yang diklasifikasikan dengan benar oleh model klasifikasi yang dibangun (Hidayatullah, *et al*., 2016).

Confussion matrix merupakan salah satu metode pengujian untuk mengetahui evaluasi akurasi terhadap sistem dalam melakukan klasifikasi sentimen. Tabel menunjukkan tabel *confussion matrix*.

Tabel 2.2 *Confussion Matrix*

		Kelas Hasil Prediksi	
		Negatif	Positif
Kelas Sebenarnya	Negatif	True Negatif (TN)	False Negatif (FN)
	Positif	False Positif (FP)	True Positif (TP)

Dari tabel tersebut dapat dilakukan perhitungan yang akan diuji dengan menghitung akurasi (*accuracy*), *recall*, precision, dan *f-measure* (Manning, et al., 2009) dengan persamaan:

Accuracy = (TN + TP)/(TN + FP + FN + TP) (2.5)

Recall = TP / (TP + FP) (2.6)

Precision = TP / (FN + TP) (2.7)

F-measure = (2 * Precision * Recall) / (Precision + Recall) (2.8)

BAB 3 METODOLOGI

Pada bab ini akan membahas tahapan-tahapan penelitian yang dilakukan yaitu klasifikasi teks Bahasa Indonesia menggunakan metode *Naïve Bayes* dan kombinasi seleksi fitur. Metode penelitian akan menjelaskan tahapan-tahapan penelitian yang dilakukan secara garis besar. Tahapan-tahapan tersebut terdiri dari tipe penelitian, strategi penelitian, peralatan pendukung, lokasi penelitian, teknik pengumpulan data, data penelitian, perancangan algoritme, implementasi sistem, pengujian sistem, hasil dan pembahasan, penutup, dan jadwal penelitian.

3.1 Tipe Penelitian

Penelitian kali ini bertipe non-implementatif. Penelitian non-implementatif merupakan penelitian yang memfokuskan pada pengamatan terhadap situasi tertentu atau fenomena yang ada. Selain itu, penelitian non-implementatif melakukan analisis terhadap relasi antara suatu fenomena yang sedang dikaji untuk memberikan hasil analisis ilmiah sebagai produk utamanya. Penelitian kali ini menghasilkan suatu produk tersebut yang diperoleh dari hasil studi kasus. Studi kasus yang diangkat pada penelitian kali ini adalah dokumen pengaduan SAMBAT Online. Pendekatan yang digunakan adalah pendekatan analitik. Pendekatan analitik merupakan pendekatan yang akan menjabarkan relasi antara elemen-elemen yang berada pada objek penelitian dengan kasus yang diteliti. Hasil akhir dari penelitian ini dapat memberikan hasil dari rumusan masalah yang sudah didefinisikan sejak awal.

3.2 Strategi Penelitian

Strategi penelitian yang dilakukan pada penelitian klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* menggunakan metode *Naïve Bayes* dan kombinasi seleksi fitur bersifat kualitatif, yaitu dengan startegi studi kasus. Studi kasus bertujuan untuk memahami objek yang diteliti. Meskipun demikian, berbeda dengan penelitian yang lain, penelitian studi kasus bertujuan secara khusus menjelaskan dan memahami objek yang ditelitinya secara khusus sebagai suatu 'kasus'. Penelitian ini difokuskan pada pengklasifikasian teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* untuk mengelompokkan dokumen pengaduan yang masuk pada SKPD yang dituju dengan otomatis.

3.3 Peralatan Pendukung

Pada tahap ini akan menjelaskan atau memaparkan tentang peralatan-peralatan pendukung sistem yang diperlukan dalam optimasi menggunakan Algoritma *Naïve Bayes* dan kombinasi seleksi fitur. Peralatan tersebut meliputi perangkat keras dan perangkat lunak. Adapun perangkat keras yang digunakan sebagai berikut:

- Intel® Core™ i3-5200U CPU @ 2.20GHz (4 CPUs)
- RAM 4GB

- HDD 500GB
- Monitor 13”

Sedangkan perangkat lunak yang digunakan yaitu:

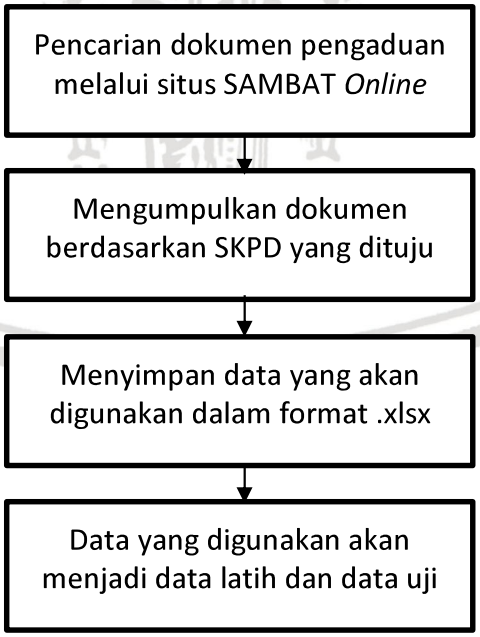
- Sistem Operasi Windows 10
- Microsoft Excel
- Python 3.6
- Spyder

3.4 Lokasi Penelitian

Penelitian kali ini yang membahas pengklasifikasian teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *online* dilaksanakan di Laboratorium Komputasi Cerdas Fakultas Ilmu Komputer Universitas Brawijaya. Penelitian ini yang dilaksanakan di laboratorium bertujuan untuk menerapkan metode klasifikasi *Naïve Bayes* dan kombinasi seleksi fitur pada pengklasifikasian teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *online*.

3.5 Teknik Pengumpulan Data

Untuk pengumpulan data pada penelitian kali ini menggunakan data primer yang diambil dari dokumen pengaduan SAMBAT *Online* yang masuk ke pemerintah Kota Malang langsung dari situs SAMBAT *Online* yang beralamat di https://sambat.malangkota.go.id/. Dokumen yang diambil berasal dari tiga SKPD yaitu DKP, DPUPPB, dan Dishub. Tahapan pengumpulan data dapat dilihat pada Gambar 3.1.



Gambar 3.1 Tahapan Pengumpulan Data

3.6 Data Penelitian

Pada penelitian kali ini, data yang digunakan adalah data teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online.* Total dokumen yang diambil adalah 204 dokumen dimana 80% dokumen akan dijadikan data latih dan 20% dokumen akan dijadikan data uji. Dari 204 dokumen terdiri dari tiga kelas yang berdasarkan SKPD yang dituju dengan pembagian 37 dokumen untuk SKPD Dinas Kebersihan dan Pertamanan (DKP), 67 dokumen untuk SKPD Dinas Pekerjaan Umum Perumahan dan Pengawasan Bangunan (DPUPPB), dan 100 dokumen untuk SKPD Dinas Perhubungan (Dishub). Dinas Kebersihan dan Pertamanan (DKP). Fitur yang digunakan adalah setiap kata yang terdapat pada dokumen latih.

3.6.1 Data Preprocessing

Data yang sudah dikumpulkan akan melalui tahap *preprocessing* terlebih dahulu agar dapat diolah. Tahapan *preprocessing* yang dilakukan ada tiga, yaitu *tokenizing/case folding, stemming,* dan *filtering/stopwords.* Pada tahapan *tokenizing/case folding* akan dilakukan proses pemecahan kalimat menjadi kata per kata dan mengubah huruf kapital menjadi huruf kecil. Kemudian pada tahapan *stemming* dilakukan pengubahan kata berimbuhan menjadi kata dasar. Pada penelitian kali ini menggunakan *stemmer* Sastrawi. Selanjutnya tahapan terakhir adalah *filtering/stopwords* dimana penghapusan kata yang terdapat pada *stoplist.* Pada penelitian kali ini menggunakan *stoplist* Tala.

3.7 Kombinasi Seleksi Fitur

Pada penelitian kali ini akan mengkombinasikan dari dua seleksi fitur yaitu *Chi-Square* dan *Information Gain.* Kombinasi dilakukan terhadap hasil dari masing-masing seleksi fitur sehingga hasil yang diberikan *Chi-Square* tidak akan mempengaruhi hasil *Information Gain* dan begitupun sebaliknya. Proses kombinasi dilakukan dengan operasi logika AND dan OR.

3.7.1 Kombinasi dengan Operasi AND

Proses kombinasi pertama yang digunakan adalah operasi logika AND. Pada proses kombinasi dengan operasi AND, apabila suatu fitur muncul di semua hasil seleksi fitur *Chi-Square* dan *Information Gain* maka fitur tersebut akan disimpan atau digunakan. Akan tetapi, apabila fitur tersebut hanya muncul di salah satu hasil seleksi fitur saja, maka fitur tersebut tidak akan digunakan dengan kata lain fitur tersebut dibuang.

3.7.2 Kombinasi dengan Operasi OR

Proses kombinasi kedua adalah dengan operasi logika OR. Pada proses kombinasi dengan operasi OR, apabila suatu fitur muncul di semua hasil seleksi fitur *Chi-Square* dan *Information Gain* maka fitur tersebut akan disimpan atau digunakan. Sedangkan jika fitur tersebut hanya muncul di salah satu hasil seleksi fitur saja, maka fitur tersebut akan tetap digunakan juga. Dengan kata lain operasi

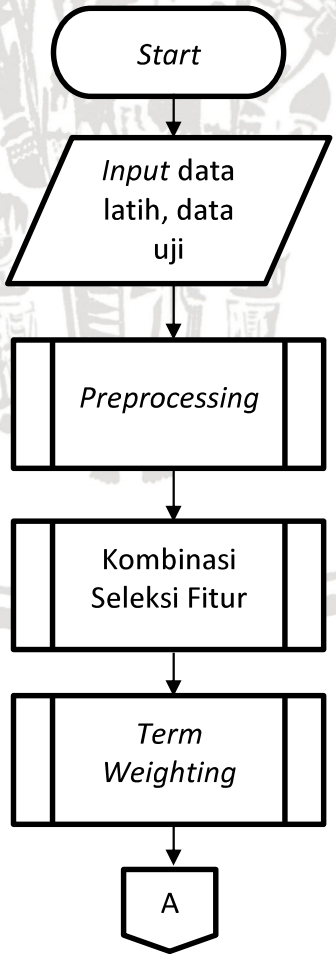
OR ini menggunakan semua fitur yang muncul pada hasil seleksi fitur *Chi-Square* dan *Information Gain.*

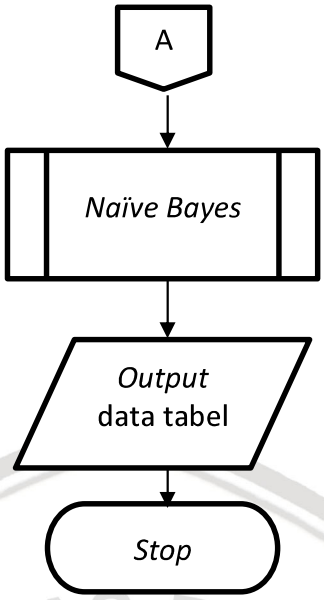
3.8 Teknik Analisis Data

Teknik analisis data pada klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* ini berupa pengujian dari hasil pengklasifikasian dokumennya dengan metode *Naïve Bayes* dan kombinasi seleksi fitur. Untuk pengujiannya menggunakan perhitungan *precission, recall*, dan *accuracy.*

3.9 Perancangan Algoritme

Pada tahap perancangan algoritme dilakukan untuk mengidentifikasi komponen-komponen yang dibutuhkan. Proses implementasi algoritme yang digunakan diawali dengan tahapan *pre-processing* pada data latih maupun data uji. Kemudian melakukan seleksi fitur dimana pada penelitian ini melakukan kombinasi seleksi fitur yaitu menggunakan *Information Gain* dan *Chi-Square*. Selanjutnya melakukan proses klasifikasi dimana pada penelitian ini menggunakan *Naïve Bayes*. Berikut tahapan-tahapan yang harus dilakukan pada penelitian ini yang dapat dilihat pada Gambar 3.2.





Gambar 3.2 Diagram Alir Proses Klasifikasi

3.10 Teknik Penerapan Metode

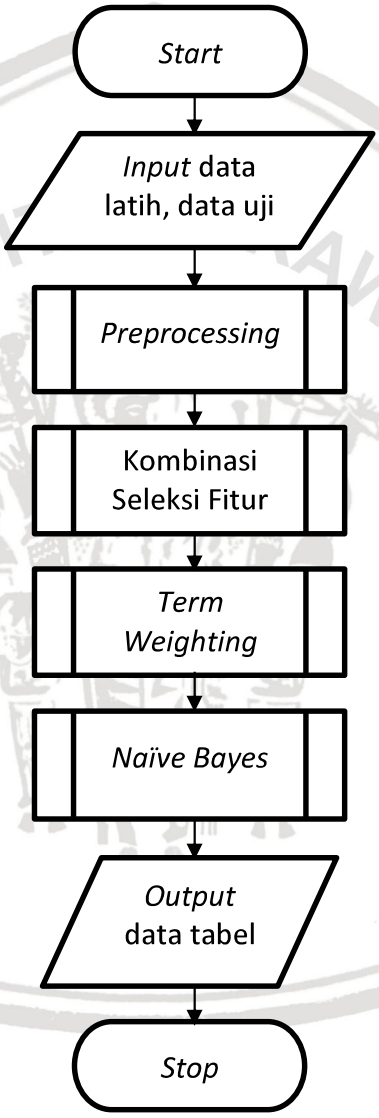
Pada teknik penerapan metode berisi teknik pendukung dalam penerapan metode yang dilakukan pada penelitian kali ini. Peneltian kali ini menggunakan *stopword list* dan proses *stemming* menggunakan library *stemmer* yaitu Sastrawi.

Implementasi adalah tahapan yang dilakukan untuk membuat sistem secara nyata yang mengacu pada perancangan sistem yang telah didefinisikan sebelumnya. Implementasi sistem mencakup:

1. Sistem yang dikembangkan menggunakan Bahasa pemrograman Python.
2. Mengumpulkan data dari situs SAMBAT *Online* dan menyimpan dalam format .csv.
3. Menerapkan metode *Naïve Bayes* dan kombinasi seleksi fitur.

BAB 4 PERANCANGAN

Pada bab ini menjelaskan tahapan-tahapan yang dilakukan dalam melakukan pengembangan sistem klasifikasi. Beberapa tahapan tersebut seperti perancangan diagram alir sistem, manualisasi dan perancangan pengujian. Alur kerja utama sistem dimulai dari *start*, *input* data latih dan uji, *preprocessing*, kombinasi seleksi fitur, *Naïve Bayes Classifier*, *output* data, *end*. Tahapan-tahapan tersebut dapat dilihat pada Gambar 4.1.



Gambar 4.1 Diagram Alir Kerja Sistem

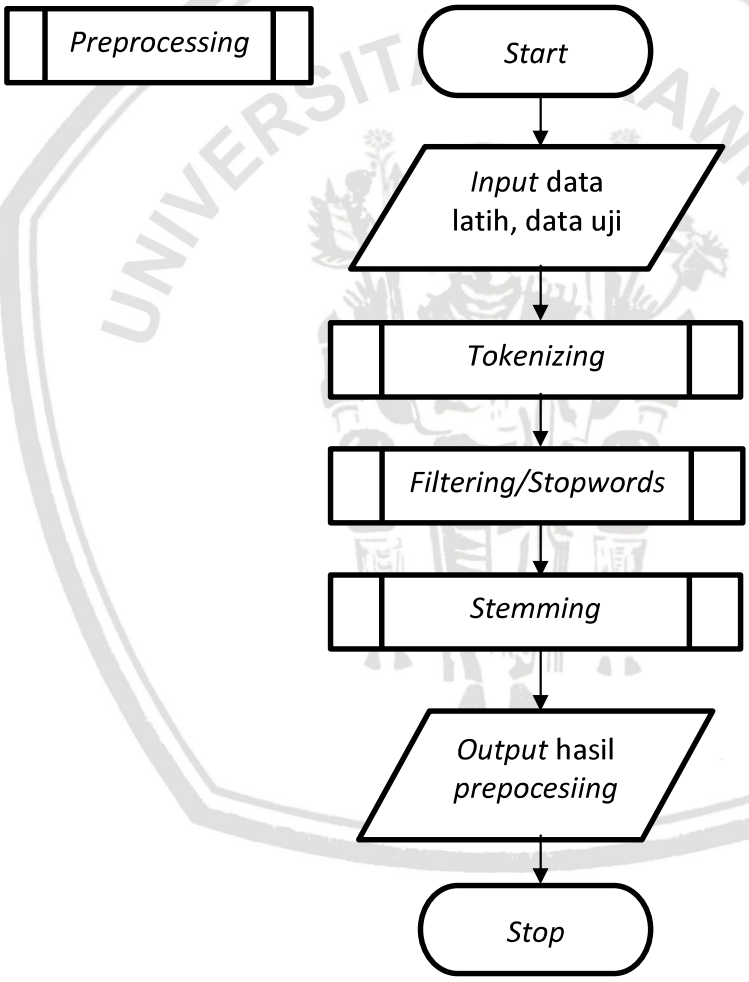
4.1 Diagram Alir Sistem (*Flowchart*)

Diagram alir sistem (*flowchart*) adalah gambaran dari tahapan-tahapan yang dilakukan dalam penyelesaian masalah. Klasifikasi teks pada sistem yang

dikembangkan kali ini dimulai dengan menginputkan data latih dan data uji yang akan digunakan. Kemudian melakukan tahapan preprocessing untuk mempersiapkan data agar dapat diolah. Selanjutnya melakukan seleksi fitur yang pada penelitian ini dilakukan kombinasi *Chi-Square* dan *Information Gain*. Kemduian tahapan klasifikiasi dimulai menggunakan metode *Naïve Bayes*. Untuk mengetahui tahapan-tahapan di atas akan dijabarkan secara lebih rinci lagi dengan diagram alir. Diagram alir tersebut dapat dilihat pada Gambar 4.2 sampai Gambar 4.17.

4.1.1 Diagram Alir *Preprocessing*

Preprocessing merupakan proses pengolahan data mentah untuk menjadi data yang siap untuk diolah. Tahapan yang dilakukan pada proses *preprocessing* meliputi *tokenizing*, *filtering/stopwords*, dan *stemming*. Diagarm alir *preprocessing* dapat dilihat pada Gambar 4.2.

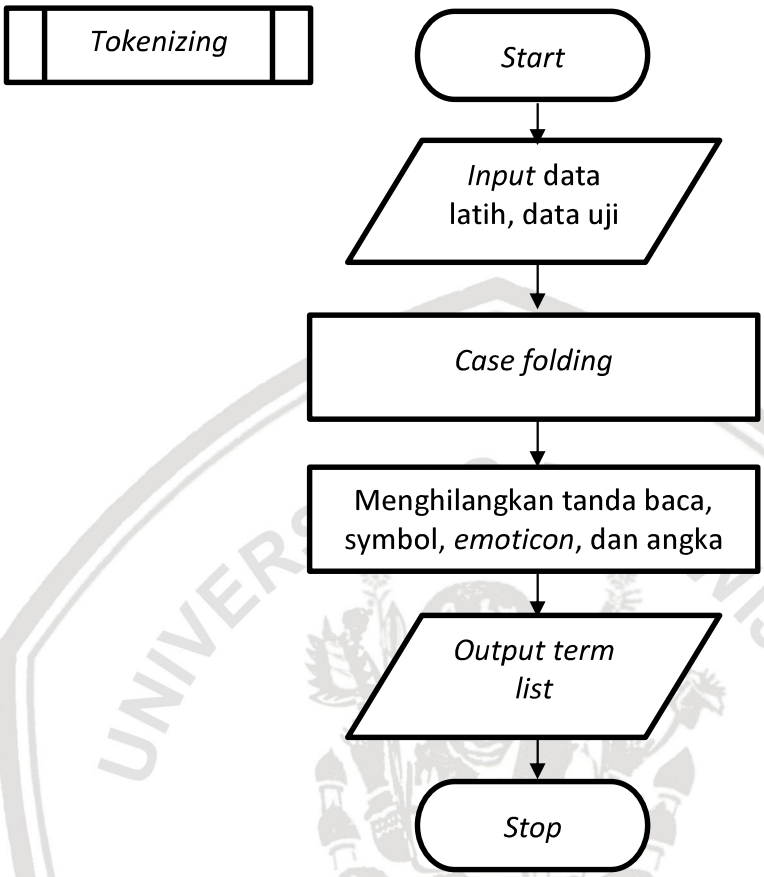


Gambar 4.2 Diagram Alir *Preprocessing*

4.1.1.1 Diagram Alir *Tokenizing*

Pada proses *tokenizing* ini mengambil kata kemudian mengubah *string* tersebut dari huruf kapital menjadi huruf kecil yang dikenal dengan proses *case folding*.

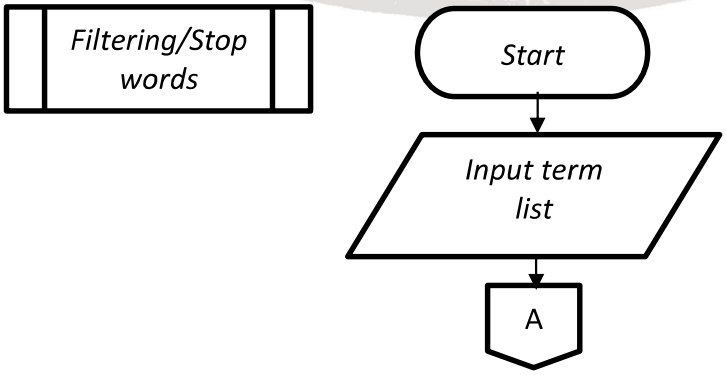
Selain itu, pada proses *tokenizing* juga menghilangkan tanda baca, *emoticon*, angka, maupun simbol. Diagram alir proses *tokenizing* dapat dilihat pada Gambar 4.3.

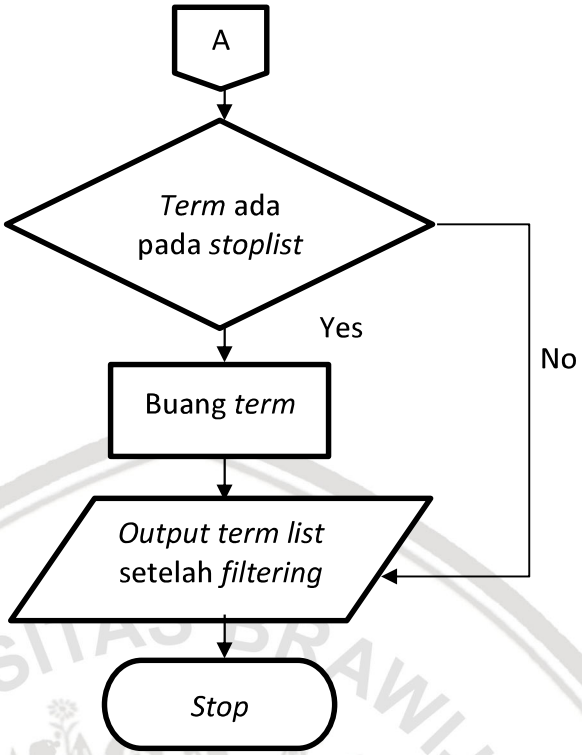


Gambar 4.3 Diagram Alir *Tokenizing*

4.1.1.2 *Filtering/Stopwords*

Pada proses *filtering/stopwords* membuang atau menghilangkan kata-kata yang tidak penting. Kata tidak penting dapat dimaksudkan dengan kata yang tidak merepresentasikan dokumen-dokumen yang ada. Kata-kata tersebut dicocokkan dengan *stoplist* untuk mengetahui apakah kata tersebut dihilangkan atau tidak. *Stoplist* biasanya berisi kata-kata penghubung atau kata ganti, misalnya "yang", "atau", "di", "dan", "saya", dan sebagainya. Diagram alir *filtering/stopwords* dapat dilihat pada Gambar 4.4.

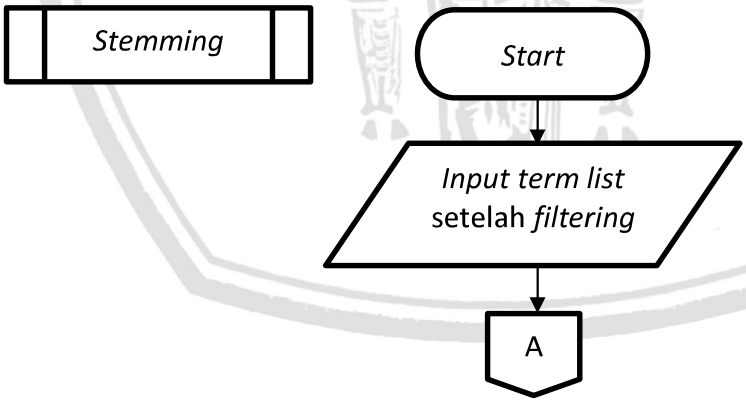


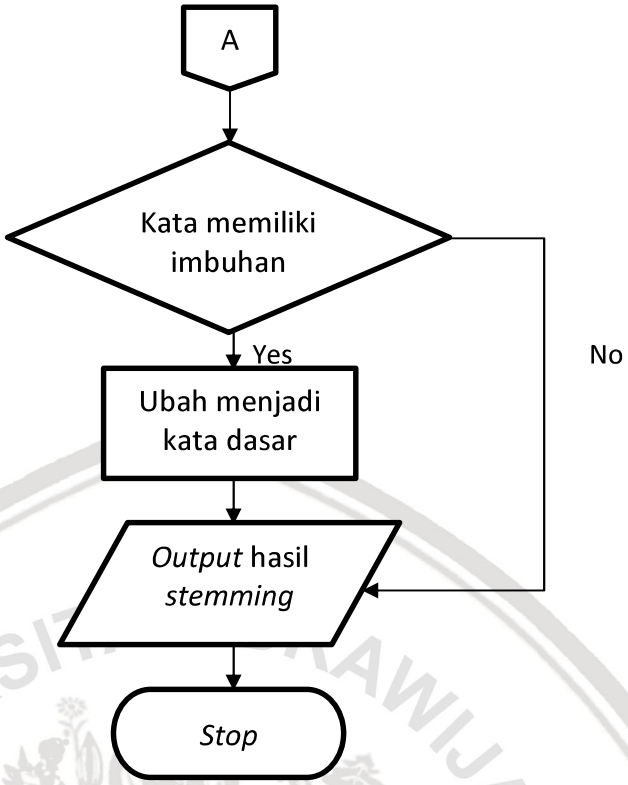


Gambar 4.4 Diagram Alir *Filtering/Stopwords*

4.1.1.3 *Diagram Alir Stemming*

Pada proses sebelumnya telah didapatkan *term list* yang diperoleh dari hasil *filtering/stopwords*. Dari kata yang ada pada *term list*, dilakukan pengubahan kata berimbuhan menjadi kata dasar atau dikenal dengan proses *stemming*. Pada proses *stemming* penelitian ini menggunakan library Sastrawi. Diagram alir proses *stemming* dapat dilihat pada Gambar 4.5.

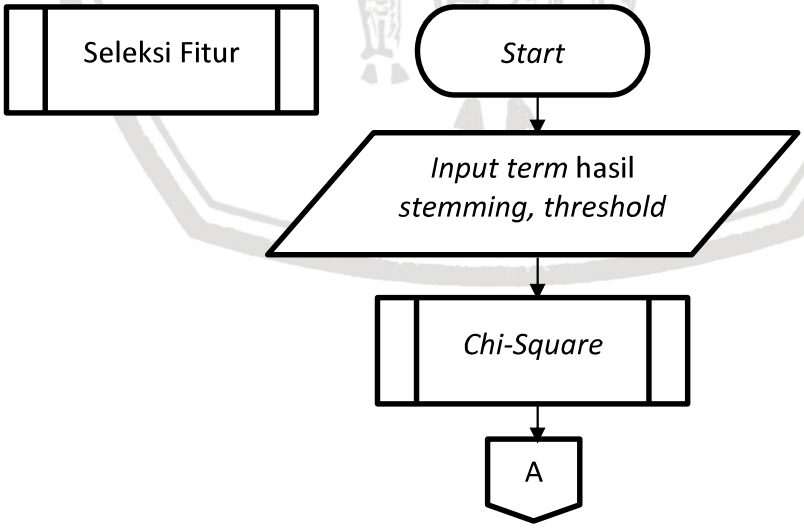


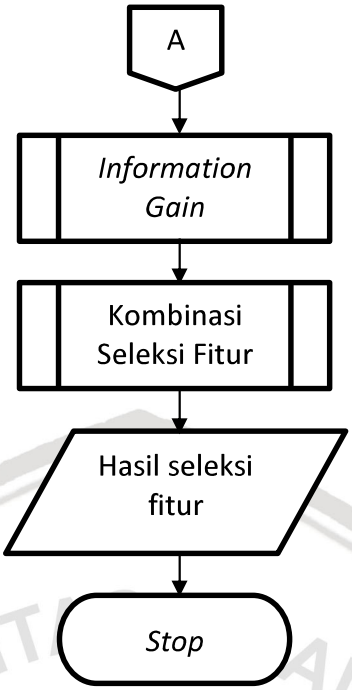


Gambar 4.5 Diagram Alir *Stemming*

4.1.2 Diagram Alir Seleksi Fitur

Pada tahap ini akan melakukan seleski fitur. Pada penelitian kali ini menggunakan dua metode seleksi fitur yaitu *Information Gain* dan *Chi-Square*. Kedua seleksi fitur tersebut akan dikombinasikan dengan operasi AND dan OR. Untuk melihat tahapan seleksi fitur yang dikombinasikan pada penelitian ini dapat dilihat pada Gambar 4.6.

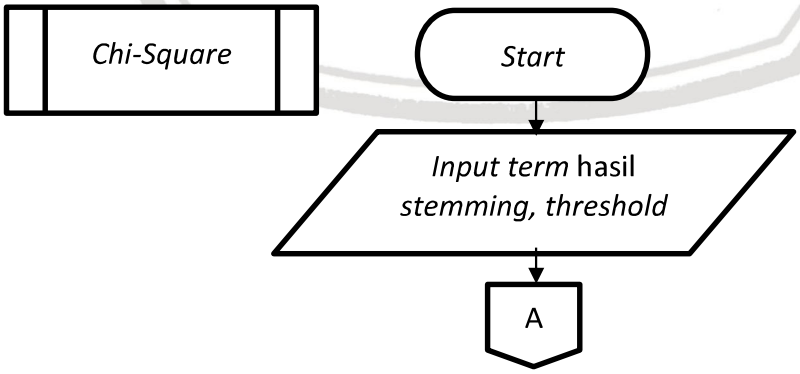


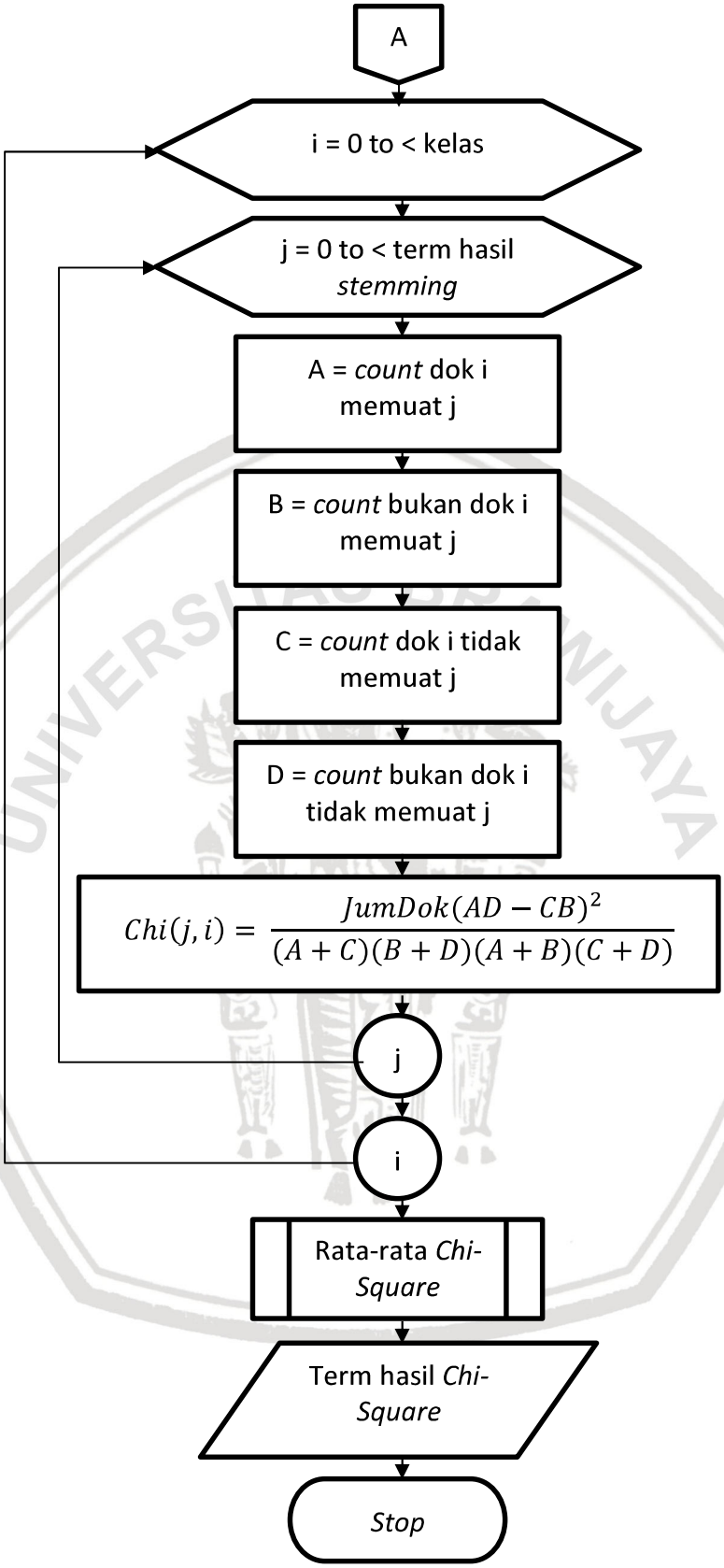


Gambar 4.6 Diagram Alir Seleksi Fitur

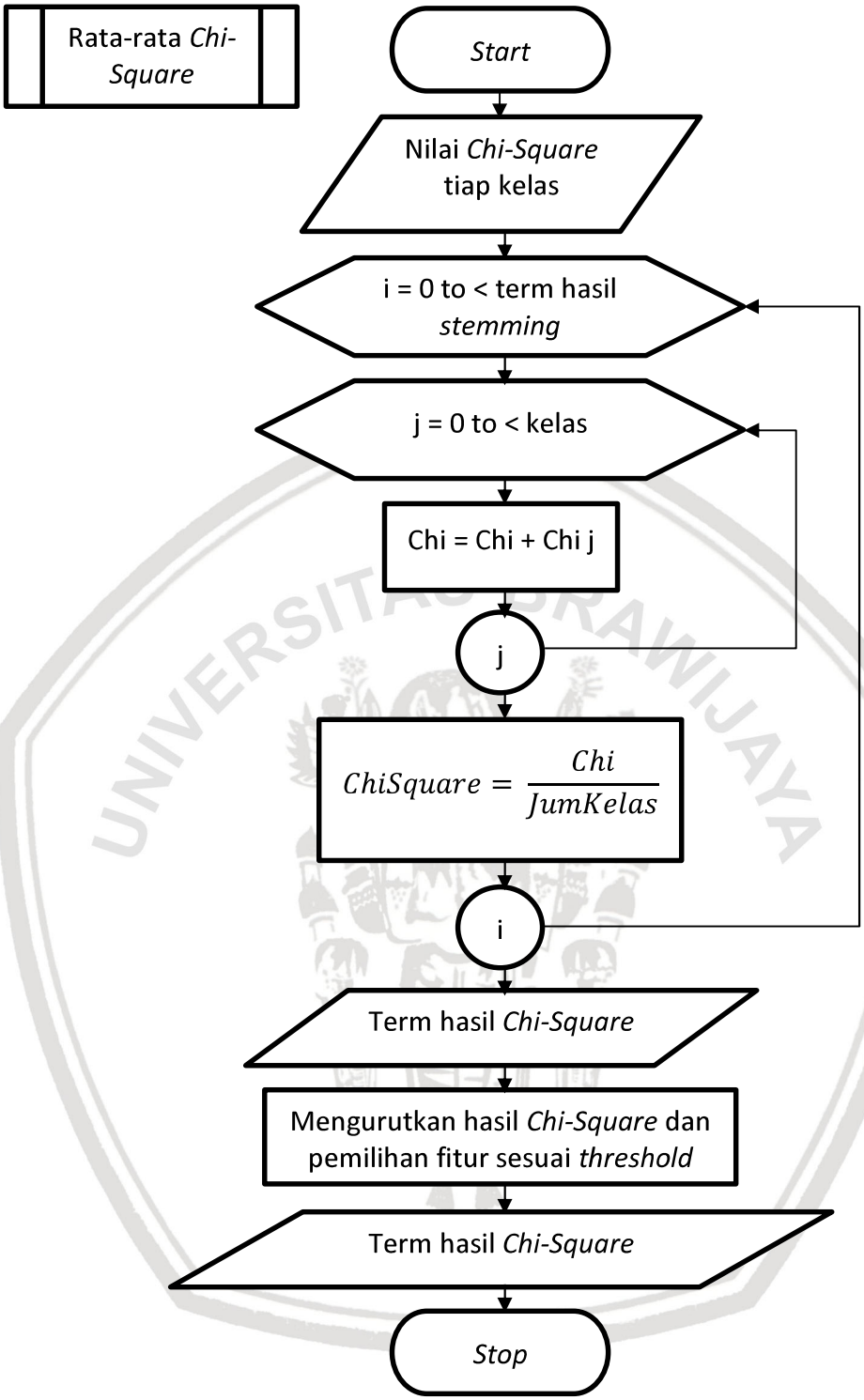
4.1.2.1 Diagram Alir *Chi-Square*

Pada tahapan seleksi fitur yang pada kasus ini menggunakan salah satunya adalah *Chi-Square* dimana terdepat 4 nilai utama yang dihitung untuk selanjutnya digunakaan saat menghitung nilai *Chi-Square*. Nilai pertama adalah menghitung nilai A yaitu jumlah dokumen pada kategori c yang membuat term. Nilai kedua adalah menghitung nilai B yaitu jumlah dokumen bukan kategori c yang memuat term. Nilai ketiga adalah menghitung nilai C yaitu jumlah dokumen c yang tidak memuat term. Nilai keempat adalah nilai D yaitu jumlah dokumen bukan c yang tidak memuat term. Setelah mendapatkan keempat nilai tersebut maka dihitunglah nilai *Chi-Square* setiap kelas sesuai dengan persamaan yang ada. Tahapan terakhir adalah mencari nilai rata-rata dari nilai *Chi-Square* setiap kelas untuk mendapatkan nilai *Chi-Square* yang sesungguhnya. Diagram alir dari proses perhitungan *Chi-Square* dapat dilihat pada Gambar 4.7.





Gambar 4.7 Diagram Alir *Chi-Square*

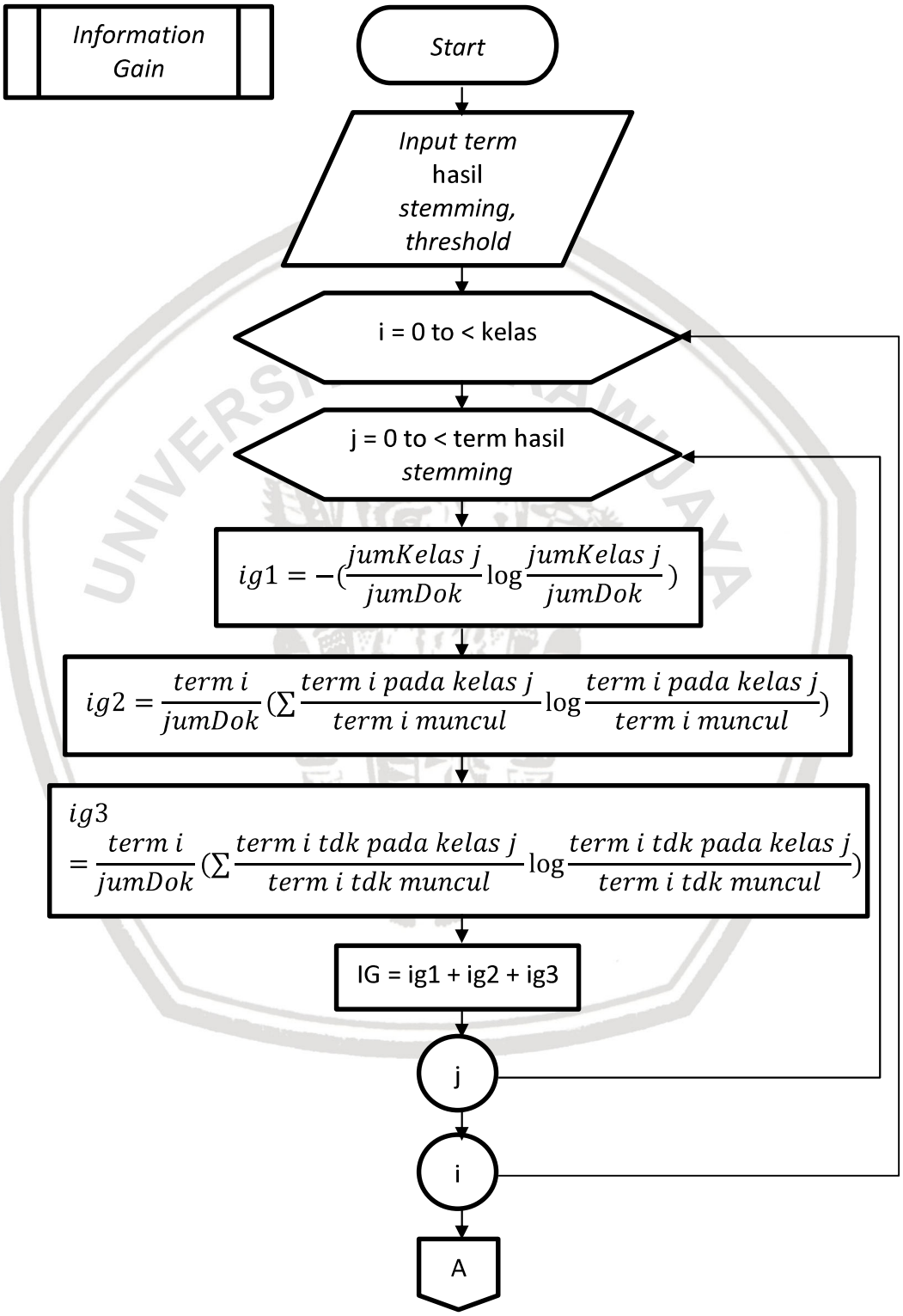


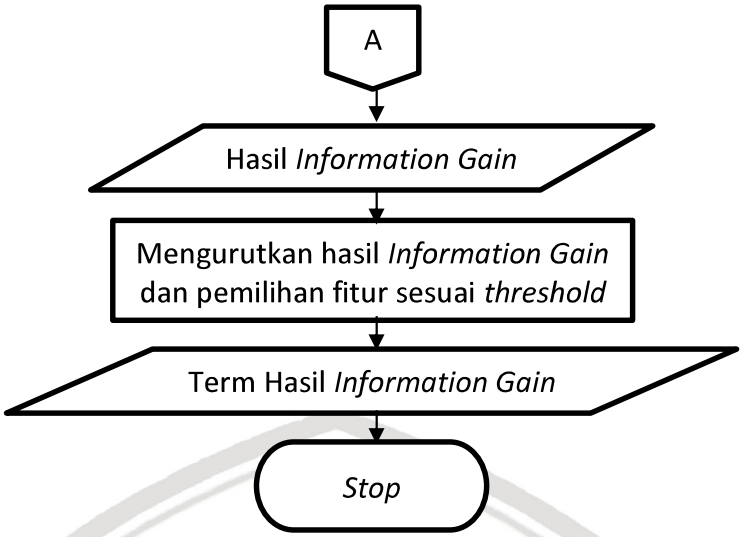
Gambar 4.8 Diagram Alir Perhitungan Rata-rata *Chi-Square*

4.1.2.2 Diagram Alir *Information Gain*

Metode seleksi fitur lain yang digunakan pada kasus ini adalah *Information Gain*. *Information Gain* dimana terdepat 4 bagian proses yang dilalui. Proses pertama adalah menghitung nilai peluang dari setiap kelas yang ada. Proses kedua adalah menghitung peluang kemunculan kata dan menghitung peluang setiap

kelas dengan syarat kata. Proses yang ketiga adalah menghitung peluang bukan kata tersebut dan menghitung peluang setiap kelas yang tidak mengandung kata tersebut. Proses yang terakhir adalah menjumlahkan semua nilai pada 3 proses sebelumnya untuk mendapatkan nilai *Information Gain*. Diagram alir dari proses perhitungan *Information Gain* dapat dilihat pada Gambar 4.9.

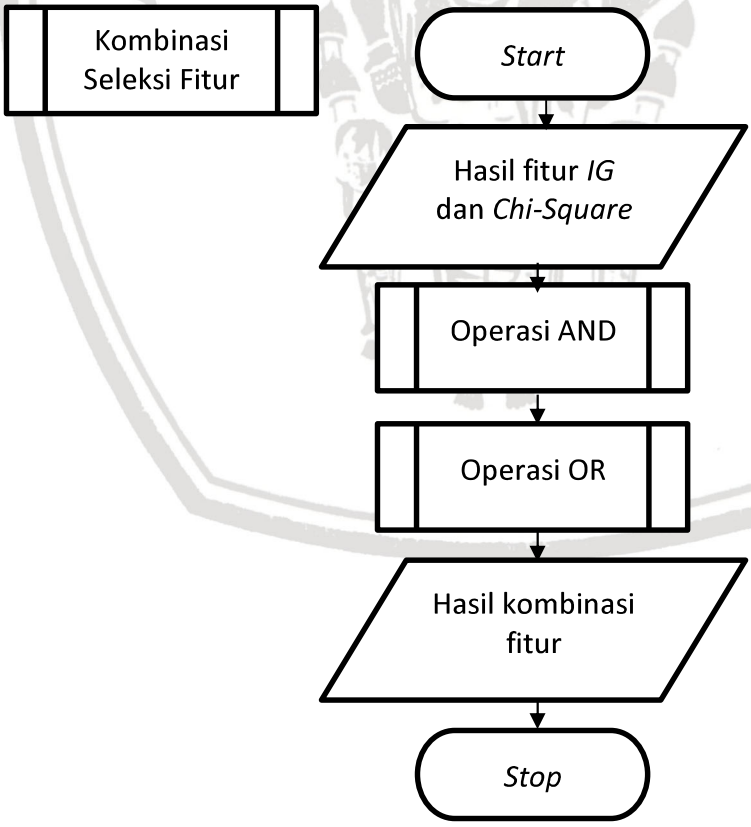




Gambar 4.9 Diagram Alir *Information Gain*

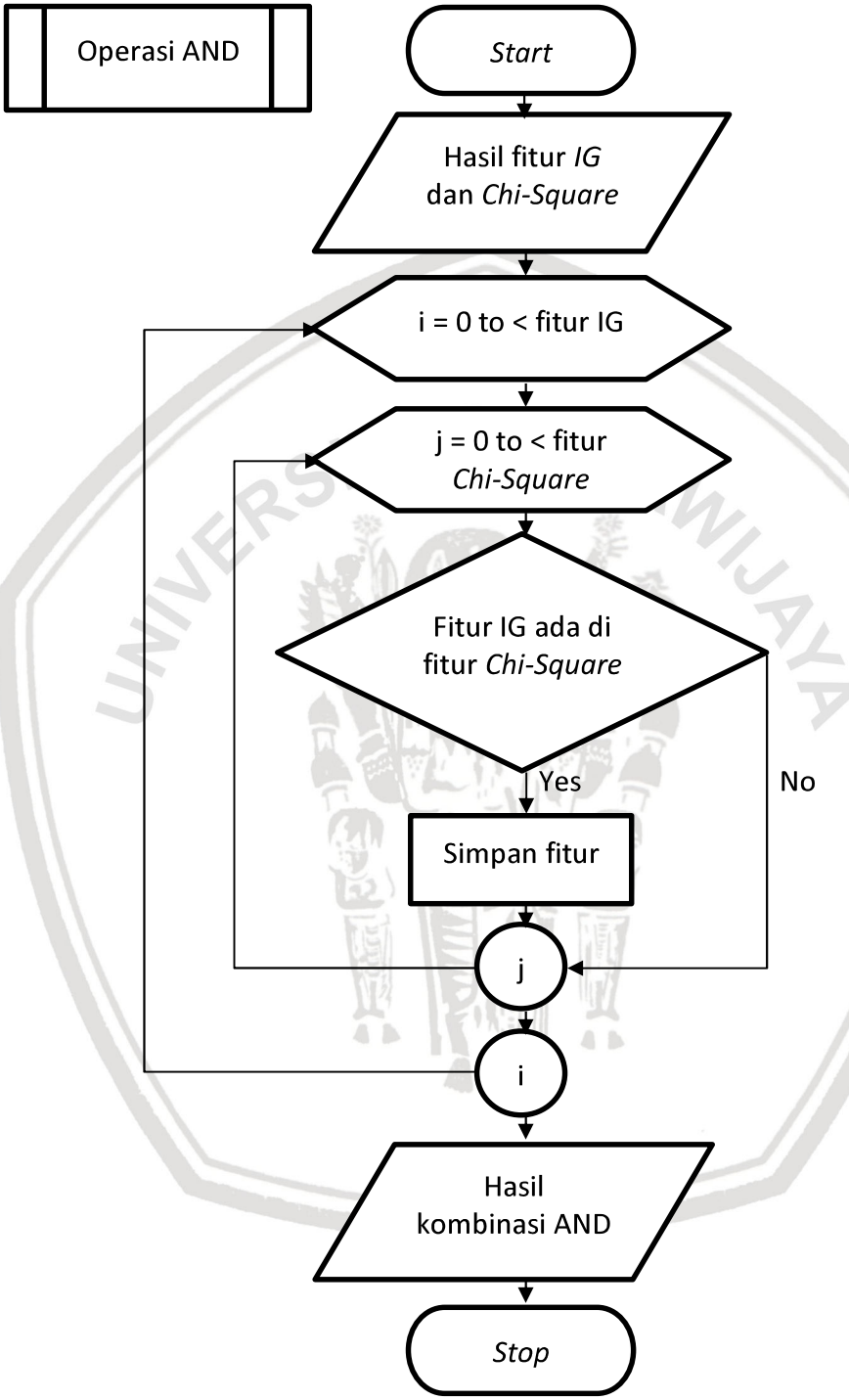
4.1.2.3 Diagram Alir Kombinasi Seleksi Fitur

Pada penelitian kali ini melakukan kombinasi seleksi fitur antara Chi-Square dan Information Gain. Proses kombinasi menggunakan dua jenis operasi yang berbeda yaitu operasi AND dan OR. Fitur hasil kombinasi akan digunakan selanjutnya pada tahapan klasifikasi. Diagram alir kombinasi seleksi fitur dapat dilihat pada Gambar 4.10.



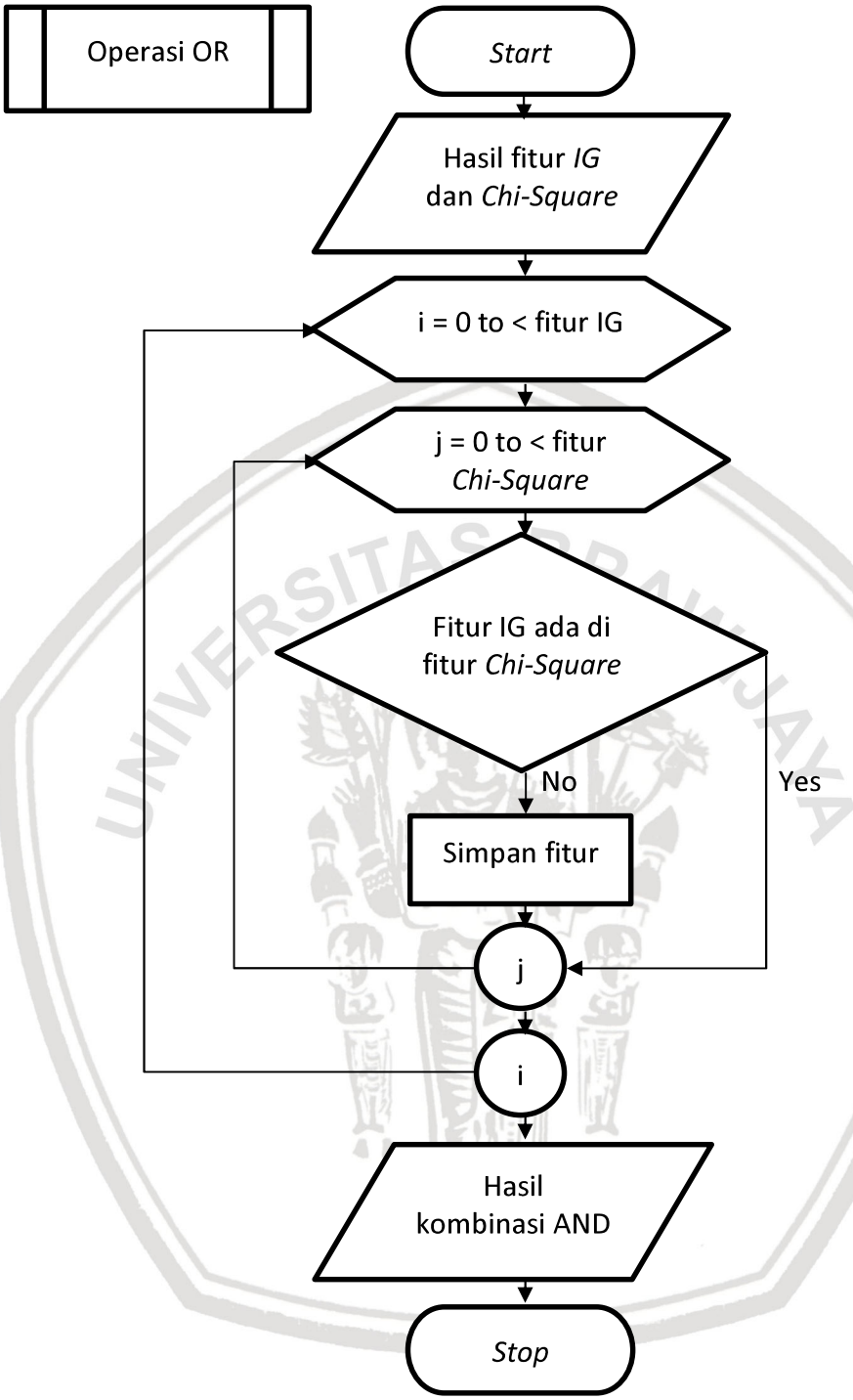
Gambar 4.10 Diagram Alir Kombinasi Seleksi Fitur

Untuk operasi AND dilakukan dengan pengecekan term yang terdapat pada kedua hasil seleksi fitur maka term akan disimpan. Apabila term tidak terdapat di salah satu hasil seleksi fitur, maka term akan dibuang. Diagram alir operasi AND dapat dilihat pada Gambar 4.11.



Gambar 4.11 Diagram Alir Kombinasi Fitur Operasi AND

Untuk operasi OR dilakukan dengan pengecekan term yang terdapat pada kedua hasil seleksi fitur atau terdapat disalah satunya maka term akan disimpan. Diagram alir operasi OR dapat dilihat pada Gambar 4.12.

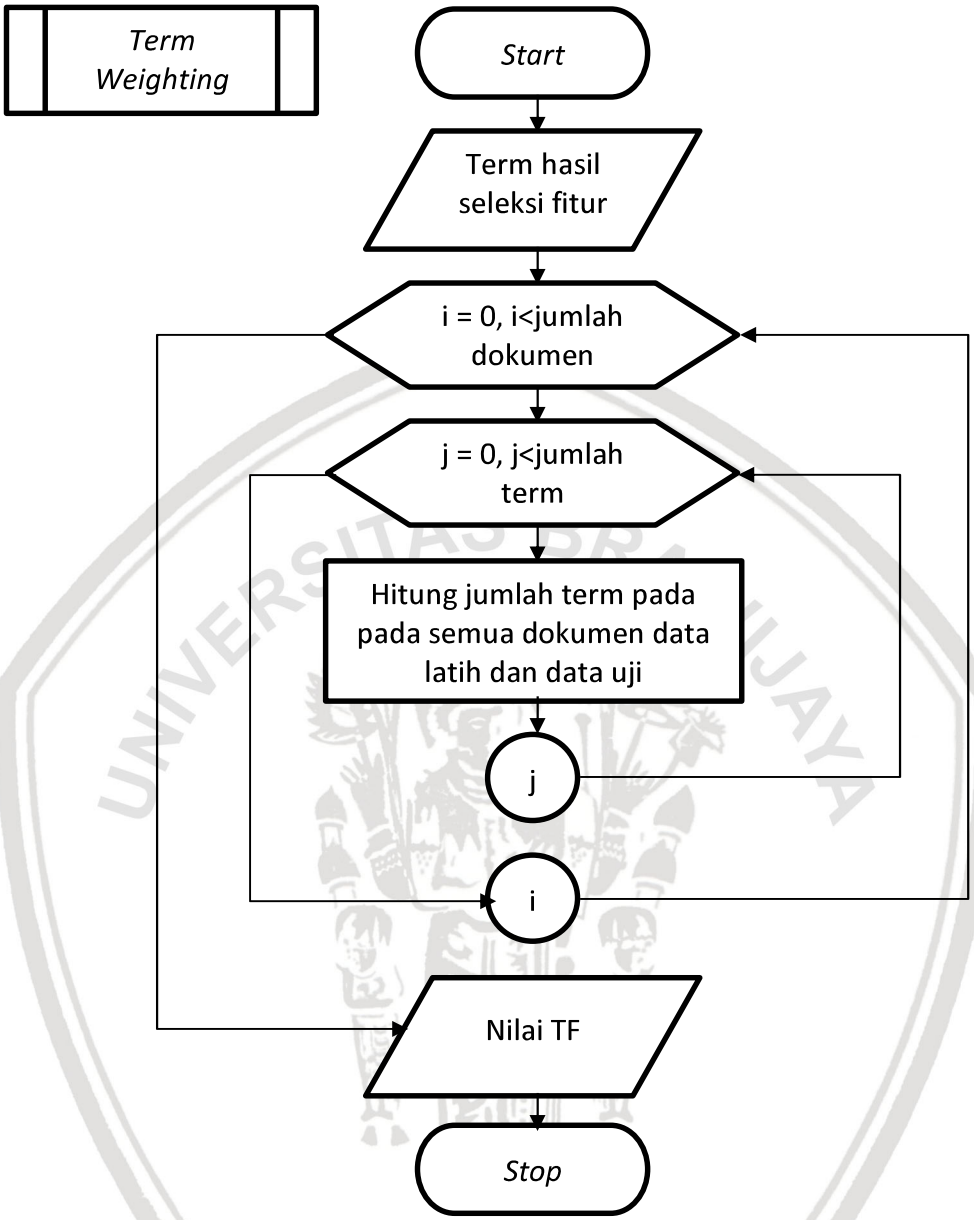


Gambar 4.12 Diagram Alir Kombinasi Fitur Operasi OR

4.1.3 Diagram Alir *Term Weighting*

Pada proses *Term Weighting* atau pembobotan kata kali ini hanya menggunakan TF (*Term Frequency*) dimana akan menghitung kemunculan setiap kata pada semua dokumen yang ada. Hal ini diperlukan untuk mempermudah

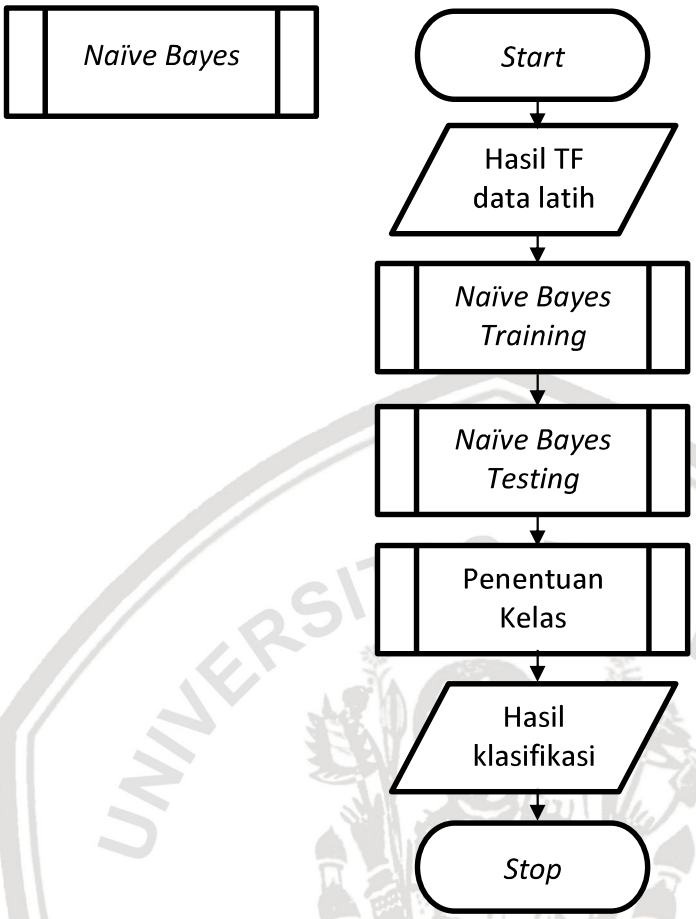
proses perhitungan klasifikasi *Naïve Bayes* nantinya. Diagram alir *Term Frequency* (TF) dapat dilihat pada Gambar 4.13.



Gambar 4.13 Diagram Alir *Term Frequency*

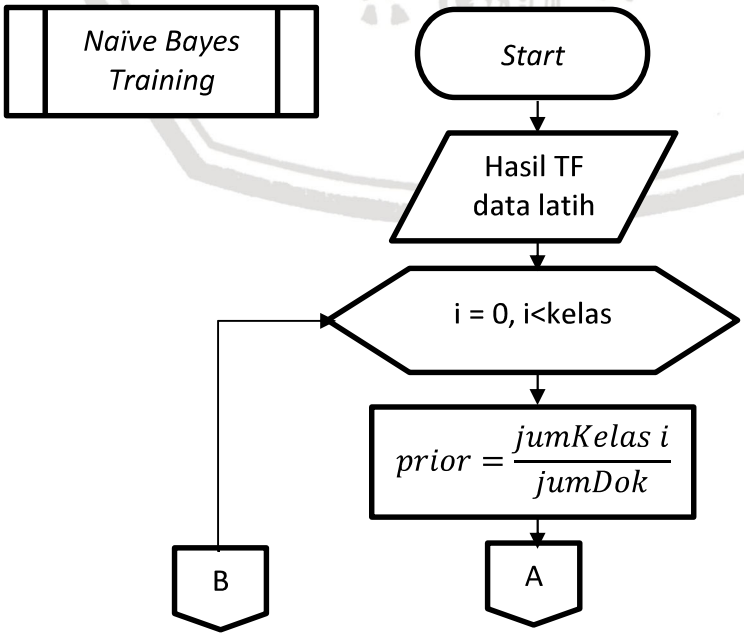
4.1.4 Diagram Alir Klasifikasi *Naïve Bayes*

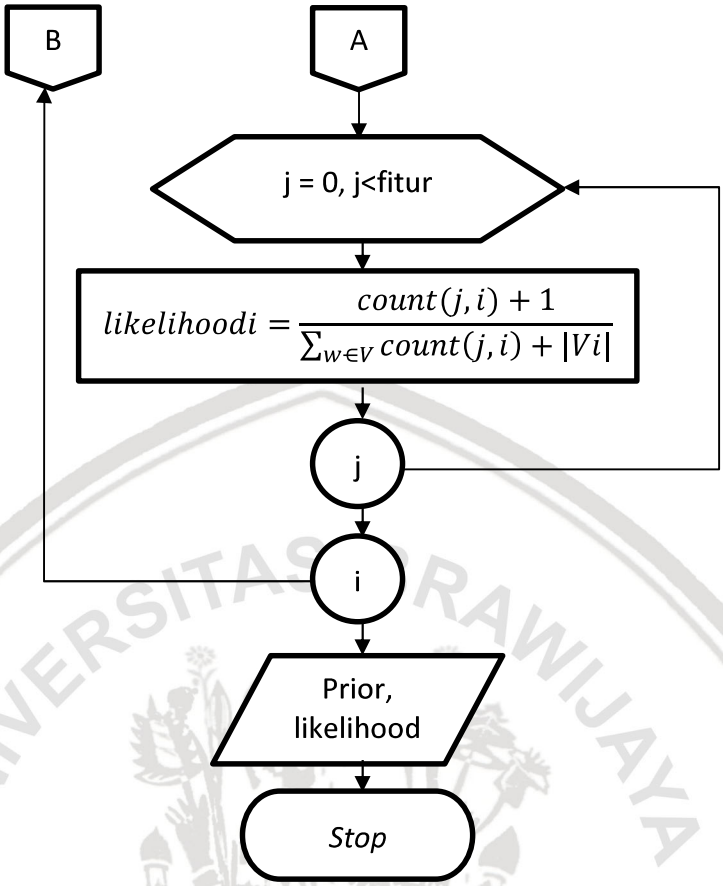
Metode klasifikasi yang digunakan pada penelitian kali ini adalah *Naïve Bayes Multinomial.* Pada perhitungan *Naïve Bayes* akan menghitung semua peluang kelas dengan syarat kata dimana untuk mendapatkan hasil itu akan melalui tiga proses. Proses pertama adalah proses training Naïve Bayes, proses kedua adalah proses *testing Naïve Bayes*, dan proses yang terakhir adalah menentukan kelas dari dokumen uji. Diagram alir perhitungan *Naïve Bayes Multinomial* dapat dilihat pada Gambar 4.14.



Gambar 4.14 Diagram Alir Naïve Bayes

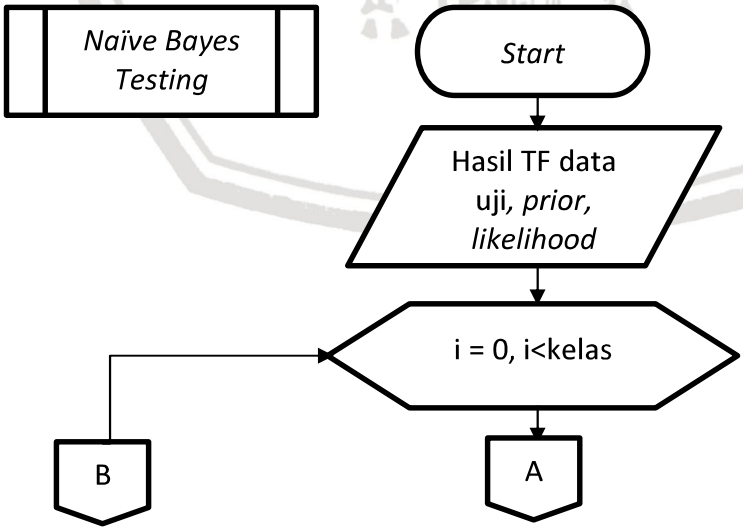
Pada proses *Naïve Bayes training* terdapat perhitungan *prior* dimana dilakukan perhitungan peluang dari semua kelas. Kemudian terdapat perhitungan *likelihood* dimana menghitung peluang dari suatu term dengan syarat kelas. Diagram alir *Naïve Bayes training* dapat dilihat pada Gambar 4.15.

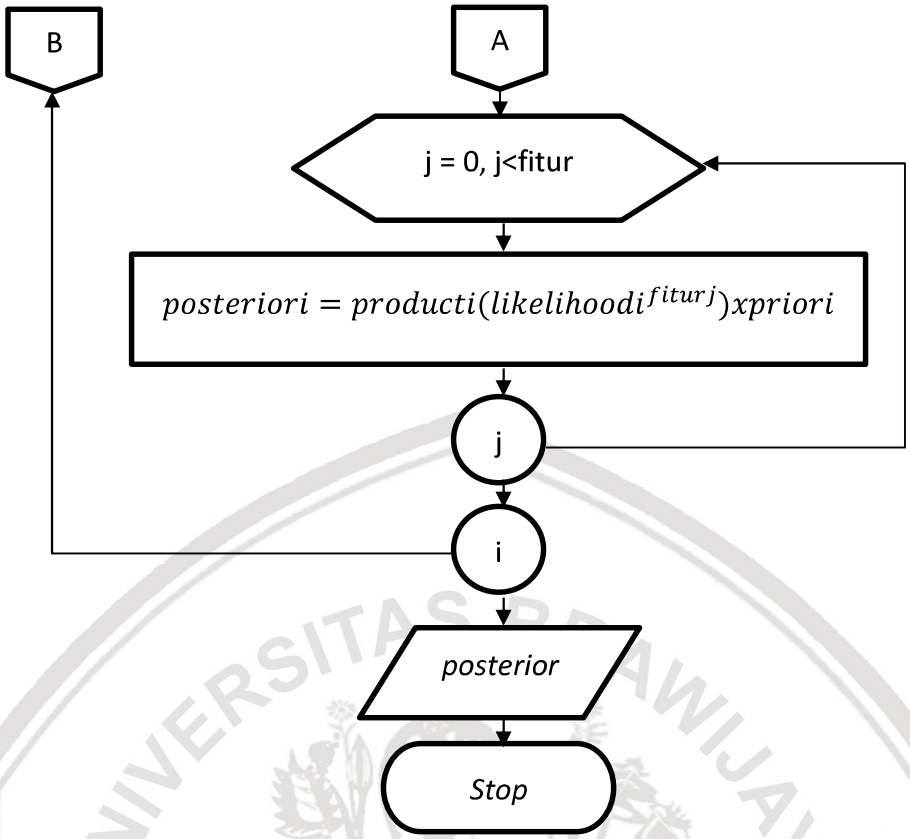




Gambar 4.15 Diagram Alir *Naïve Bayes Training*

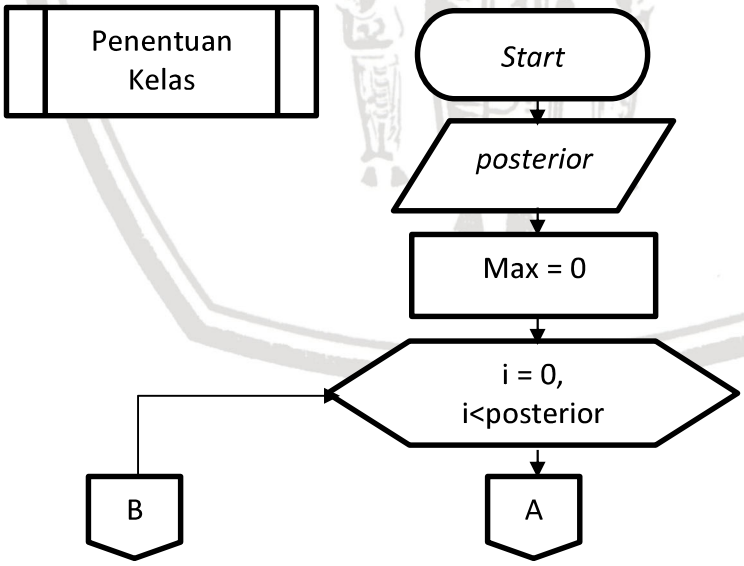
Pada proses *Naïve Bayes testing* terdapat perhitungan *posterior* dimana dilakukan perhitungan peluang kelas dengan syarat kata. Perhitungan posterior dengan mengalikan prior dengan likelihood yang sudah dipangkatkan sesuai dengan jumlah kemunculan kata data uji. Diagram alir *Naïve Bayes testing* dapat dilihat pada Gambar 4.16.

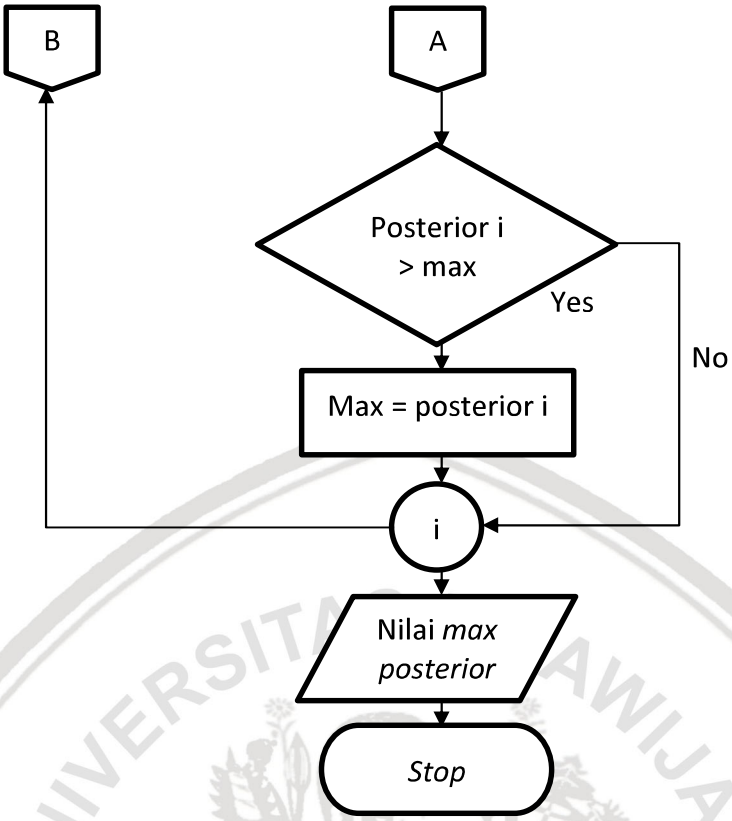




Gambar 4.16 Diagram Alir *Naïve Bayes Testing*

Setelah mendapatkan nilai posterior maka selanjutnya akan dibandingkan nilai posterior setiap kelas. Kelas yang memiliki nilai posterior terbesar maka kelas tersebut terpilih menjadi kelas terpilih. Diagram alir penentuan kelas dapat dilihat pada Gambar 4.17.





Gambar 4.17 Diagram Alir Penentuan Kelas

4.2 Manualisasi Perhitungan Data

Manualisasi perhitungan data adalah proses dimana melukan perhitungan manual pada penerapan kombinasi seleksi fitur yang menggunakan *information Gain* dan *Chi-Square* untuk menentukan fitur penting untuk digunakan pada penelitian ini. Selain itu akan melakukan perhitungan manual juga pada proses klasifikasi untuk mengetahui hasil klasifikasi termasuk ke dalam kelas SKPD mana yang dituju. Data latih yang digunakan dapat dilihat pada Tabel 4.1:

Tabel 4.1 Data Latih

No.	Isi Dokumen	SKPD
1	Pasar Madyopuro sudah lama tidak ada karcis parkir.	Dishub
2	Banyak tukang parkir tidak memberikan karcis.	Dishub
3	Mohon ditindak lanjuti di wilayah kelurahan kauman petugas membuang sampahnya disungai.	DKP
4	Jembatan Muharto bukan tps tapi selalu banyak sampah	DKP
5	Mohon perbaikan akses jalan publik depan perumahan Sukun Pondok Indah.	DPUPPB
6	Dinoyo banyak aspal rusak harap perbaikannya	DPUPPB

Data uji adalah data yang akan diuji masuk ke kelas mana dari ketiga kelas SKPD yang ada. Data uji yang digunakan dapat dilihat pada Tabel 4.2.

Tabel 4.2 Data Uji

No.	Isi Dokumen	SKPD
1	Karcis parkir tidak ada di Indomaret jalan kawi	?

4.2.1 Preprocessing

Tahapan pertama yang dilakukan adalah *preprocessing*. *Preprocessing* terdapat beberapa tahapan. Tahapan pertama adalah melakukan tokenisasi memecah kalimat menjadi kata, menghapus karakter maupun angkta, dan dilakukan *case folding* dengan mengubah semua kata menjadi huruf kecil. Kemudian dilakukan proses *filtering/stopword* dengan menghapus kata-kata yang tidak penting pada dokumen sesuai dengan *stoplist* yang ada. Tahapan terakhir adalah *stemming* dimana akan mengubah kata berimbuhan menjadi kata dasar.

4.2.1.1 *Tokenizing* dan *Case Folding*

Pada tahapan *tokenizing* ini akan memecah kalimat menjadi kata tunggal atau bisa disebut dengan *term*. Pada tahapan ini juga akan menghilangkan semua karakter, simbol, maupun angka atau apapun selain *alphabet*. Dilakukan juga *case folding* yaitu mengubah semua huruf kapital menjadi huruf kecil. Untuk hasil *tokenizing* dan *case folding* data latih dapat dilihat pada Tabel 4.3.

Tabel 4.3 Hasil *Tokenizing* dan *Case Folding* Data Latih

No.	Isi Dokumen	SKPD
1	pasar | madyopuro | sudah | lama | tidak | ada | karcis | parkir	Dishub
2	banyak | tukang | parkir | tidak | memberikan | karcis	Dishub
3	mohon | ditindak | lanjuti | di | wilayah | kelurahan | kauman | petugas | membuang | sampahnya | disungai	DKP
4	jembatan | muharto | bukan | tps | tapi | selalu | banyak | sampah	DKP
5	mohon | perbaikan | akses | jalan | publik | depan | perumahan | sukun | pondok | indah	DPUPPB
6	dinoyo | banyak | aspal | rusak | harap | perbaikannya	DPUPPB

Tabel di atas merupakan hasil tokenisasi dari data latih. Terlihat kalimat sudah terpecah menjadi kata yang berdiri sendiri. Kemudian dilakukan juga hal yang sama pada data uji. Hasil tokenisasi dan *case folding* data uji dapat dilihat pada Tabel 4.4.

Tabel 4.4 Hasil *Tokenizing* dan *Case Folding* Data Uji

No.	Isi Dokumen	SKPD
1	karcis | parkir | tidak | ada | di | Indomaret | jalan | kawi	?

4.2.1.2 *Filtering/Stopword*

Filtering/stopwords merupakan tahapan dimana dilakukan pengecakan setiap *term* yang ada. Kemudian setiap *term* akan dicek pada *stoplist* yang sudah ada. Jika terdapat dapat *stoplist* maka *term* tersebut dianggap tidak penting. *Term* yang tidak penting akan dibuang sedangkan *term* yang penting akan teruse digunakan. Untuk hasil dari *filtering/stopwords* data latih dapat dilihat pada Tabel 4.5.

Tabel 4.5 Hasil *Filtering/Stopwords* Data Latih

No.	Isi Dokumen	SKPD
1	pasar | madyopuro | karcis | parkir	Dishub
2	tukang | parkir | memberikan | karcis	Dishub
3	mohon | ditindak | wilayah | kelurahan | kauman | petugas | membuang | sampahnya | disungai	DKP
4	jembatan | muharto | tps | sampah	DKP
5	mohon | perbaikan | akses | jalan | publik | perumahan | sukun | pondok | indah	DPUPPB
6	dinoyo | aspal | rusak | harap | perbaikannya	DPUPPB

Pada tabel di atas terlihat beberapa kata yang hilang dari proses sebelumnya dikarenakan kata yang terdapat pada *stoplist* telah dibuang. Selanjutnya dilakukan hal serupa pada data uji. Hasil dari *filtering/stopwords* data latih dapat dilihat pada Tabel 4.6.

Tabel 4.6 Hasil *Filtering/Stopwords* Data Uji

No.	Isi Dokumen	SKPD
1	karcis | parkir | Indomaret | jalan | kawi	?

4.2.1.3 *Stemming*

Tahapan *stemming* merupakan tahapan dimana mengubah sebuah kata berimbuhan menjadi sebuah kata dasar. Proses yang dilakukan dengan menghilangkan imbuhan baik di depan maupun di belakang. Hasil *stemming* data latih dapat dilihat pada Tabel 4.7.

Tabel 4.7 Hasil *Stemming* Data Latih

No.	Isi Dokumen	SKPD
1	pasar | madyopuro | karcis | parkir	Dishub
2	tukang | parkir | beri | karcis	Dishub
3	mohon | tindak | wilayah | lurah | kauman | tugas | buang | sampah | sungai	DKP
4	jembatan | muharto | tps | sampah	DKP

No.	Isi Dokumen	SKPD
5	mohon | baik | akses | jalan | publik | rumah | sukun | pondok | indah	DPUPPB
6	dinoyo | aspal | rusak | harap | baik	DPUPPB

Tabel di atas dapat dilihat setiap kata yang berimbuhan telah diubah menjadi kata dasar. Hal serupa juga dilakukan pada data uji. Hasil *stemming* data uji dapat dilihat pada Tabel 4.8.

Tabel 4.8 Hasil *Stemming* Data Uji

No.	Isi Dokumen	SKPD
1	karcis | parkir | Indomaret | jalan | kawi	?

4.2.2 Seleksi Fitur

Seleksi fitur merupakan proses dimana menyeleksi fitur yang akan digunakan pada tahapan klasifikasi. Seleksi fitur dilakukan dengan cara memilih fitur-fitur yang relevan yang mempengaruhi hasil klasifikasi. Seleksi fitur digunakan untuk mengurangi dimensi data dan fitur-fitur yang tidak relevan. Seleksi fitur digunakan untuk meningkatkan efektifitas dan efisiensi kinerja dari algoritme klasifikasi. Pada penelitian kali ini menggunakan *Information Gain* dan *Chi-Square* yang dikombinasikan.

4.2.2.1 *Information Gain*

Proses perhitungan nilai *Information Gain* adalah dengan menghitung nilai peluang setiap kategori pada kasus ini peluang tiap SKPD. Selain itu menghitung peluang kata dan peluang ada atau tidak adanya kata yang terdapat pada setiap kategori yang ada. Hasil dari setiap peluang tadi akan dijumlah sesuai dengan persamaan yang ada dan hasilnya adalah nilai *Information Gain*. Contoh *term* yang akan digunakan pada perhitungan *Information Gain* adalah "parkir" yang dapat dilihat pada Tabel 4.9.

Tabel 4.9 Term Parkir

No.	Term	Dok 1	Dok 2	Dok 3	Dok 4	Dok 5	Dok 6
1	Parkir	1	1	0	0	0	0

Di bawah ini adalah contoh proses perhitungan *Information Gain* berdasarkan Persamaan 2.1 pada salah satu term yang digunakan yaitu "parkir". Pada persamaan 2.1 dimana *ci* adalah peluang setiap kelas, $ci|t$ adalah peluang term tersebut ada pada setiap kelas, dan $ci|\bar{t}$ adalah peluang tidak munculnya term pada kelas.

$$IG(t) = -(\sum_{i=1}^{|c|} P(ci)\log P(ci) + P(t)\sum_{i=1}^{|c|} P(ci|t)\log P(ci|t)$$
$$+ P(\bar{t})\sum_{i=1}^{|c|} P(ci|\bar{t})\log P(ci|\bar{t})$$
$$= -\left(\frac{2}{6}log\frac{2}{6}+\frac{2}{6}\log\frac{2}{6}+\frac{2}{6}log\frac{2}{6}\right)+\left(\left(\frac{2}{6}\right)\frac{2}{2}log\frac{2}{2}+\frac{0}{2}log\frac{0}{2}+\frac{0}{2}log\frac{0}{2}\right)$$
$$+\left(\left(\frac{4}{6}\right)\frac{0}{4}log\frac{0}{4}+\frac{2}{4}log\frac{2}{4}+\frac{2}{4}log\frac{2}{4}\right)$$
$$= -(-0{,}477121)+(0)+(-0{,}2006686)$$
$$= 0{,}276455$$

Berikut adalah hasil perhitungan dari nilai *information gain* seluruh term pada data *training* yang dapat dilihat pada Tabel 4.10.

Tabel 4.10 Hasil *Information Gain* Data Latih

No	Term	Dok1	Dok2	Dok3	Dok4	Dok5	Dok6	*Information Gain*
1	pasar	1	0	0	0	0	0	0,095333
2	madyopuro	1	0	0	0	0	0	0,095333
3	karcis	1	1	0	0	0	0	0,276435
4	parkir	1	1	0	0	0	0	0,276435
5	tukang	0	1	0	0	0	0	0,095333
6	Beri	0	1	0	0	0	0	0,095333
7	mohon	0	0	1	0	1	0	0,075748
8	tindak	0	0	1	0	0	0	0,095333
9	wilayah	0	0	1	0	0	0	0,095333
10	lurah	0	0	1	0	0	0	0,095333
11	kauman	0	0	1	0	0	0	0,095333
12	tugas	0	0	1	0	0	0	0,095333
13	buang	0	0	1	0	0	0	0,095333
14	sampah	0	0	1	1	0	0	0,276435
15	sungai	0	0	1	0	0	0	0,095333
16	jembatan	0	0	0	1	0	0	0,095333
17	muharto	0	0	0	1	0	0	0,095333
18	Tps	0	0	0	1	0	0	0,095333

No	Term	Dok1	Dok2	Dok3	Dok4	Dok5	Dok6	*Information Gain*
19	Baik	0	0	0	0	1	1	0,276435
20	akses	0	0	0	0	1	0	0,095333
21	jalan	0	0	0	0	1	0	0,095333
22	publik	0	0	0	0	1	0	0,095333
23	rumah	0	0	0	0	1	0	0,095333
24	sukun	0	0	0	0	1	0	0,095333
25	pondok	0	0	0	0	1	0	0,095333
26	indah	0	0	0	0	1	0	0,095333
27	dinoyo	0	0	0	0	0	1	0,095333
28	aspal	0	0	0	0	0	1	0,095333
29	rusak	0	0	0	0	0	1	0,095333
30	harap	0	0	0	0	0	1	0,095333

4.2.2.2 Pengurutan Term Hasil *Inforamtion Gain*

Setelah mendapatkan nilai *information gain* setiap term, selanjutnya term tersebut akan diurutkan dari term yang memiliki nilai *information gain* tertinggi hingga terendah. Untuk contoh ini akan mengambil fitur sebanyak 75% dari hasil *information gain* yaitu 23 term dari 30 term. Pengurutan term dan fitur yang diambil dapat dilihat pada Tabel 4.11.

Tabel 4.11 Hasil Pengurutan Term dari *Information Gain*

No	Term	Dok1	Dok2	Dok3	Dok4	Dok5	Dok6	*Information Gain*
1	karcis	1	1	0	0	0	0	0,276435
2	parkir	1	1	0	0	0	0	0,276435
3	sampah	0	0	1	1	0	0	0,276435
4	Baik	0	0	0	0	1	1	0,276435
5	pasar	1	0	0	0	0	0	0,095333
6	madyopuro	1	0	0	0	0	0	0,095333
7	tukang	0	1	0	0	0	0	0,095333
8	Beri	0	1	0	0	0	0	0,095333
9	tindak	0	0	1	0	0	0	0,095333
10	wilayah	0	0	1	0	0	0	0,095333

No	Term	Dok1	Dok2	Dok3	Dok4	Dok5	Dok6	*Information Gain*
11	lurah	0	0	1	0	0	0	0,095333
12	kauman	0	0	1	0	0	0	0,095333
13	tugas	0	0	1	0	0	0	0,095333
14	buang	0	0	1	0	0	0	0,095333
15	sungai	0	0	1	0	0	0	0,095333
16	jembatan	0	0	0	1	0	0	0,095333
17	muharto	0	0	0	1	0	0	0,095333
18	Tps	0	0	0	1	0	0	0,095333
19	akses	0	0	0	0	1	0	0,095333
20	jalan	0	0	0	0	1	0	0,095333
21	publik	0	0	0	0	1	0	0,095333
22	rumah	0	0	0	0	1	0	0,095333
23	sukun	0	0	0	0	1	0	0,095333
24	pondok	0	0	0	0	1	0	0,095333
25	indah	0	0	0	0	1	0	0,095333
26	dinoyo	0	0	0	0	0	1	0,095333
27	aspal	0	0	0	0	0	1	0,095333
28	rusak	0	0	0	0	0	1	0,095333
29	harap	0	0	0	0	0	1	0,095333
30	mohon	0	0	1	0	1	0	0,075748

Setelah term diurutkan maka diambil sesuai persentasi jumlah fitur yang akan diambil. Pada contoh kali ini mengektraksi fitur sebanyak 75% dari 30 fitur yaitu 23 fitur. Untuk term yang terpilih dari *Information Gain* dapat dilihat pada Tabel 4.12.

Tabel 4.12 Term Terpilih dari *Information Gain*

No.	Term
1	karcis
2	parkir
3	sampah
4	baik

No.	Term
5	pasar
6	madyopuro
7	tukang
8	beri
9	tindak
10	wilayah
11	lurah
12	kauman
13	tugas
14	buang
15	sungai
16	jembatan
17	muharto
18	tps
19	akses
20	jalan
21	publik
22	rumah
23	sukun

4.2.2.3 *Chi-Square*

Chi-Square merupakan salah satu seleksi fitur dengan menghitung jumlah dokumen pada setiap kelas yang mengandung term tertentu, jumlah dokumen yang bukan kelas tersebut yang mengandung term tersebut, jumlah dokumen pada kelas tersebut yang tidak mengandung term tersebut dan jumlah dokumen yang bukan kelas tersebut yang tidak mengandung term tersebut. Nilai *Chi-Square* setiap kelas akan dirata-ratakan untuk mendapatkan nilai *Chi-Square* dari setiap term. Contoh *term* yang akan digunakan pada perhitungan *Chi-Square* adalah "karcis" yang dapat dilihat pada Tabel 4.13.

Tabel 4.13 Term Karcis

No.	Term	Dok 1	Dok 2	Dok 3	Dok 4	Dok 5	Dok 6
1	Karcis	1	1	0	0	0	0

Di bawah ini adalah contoh proses perhitungan *Chi-Square* berdasarkan Persamaan 2.2 pada salah satu term yang digunakan yaitu "karcis". Pada

Persamaan 2.2 dimana N adalah jumlah dokumen, t adalah term, dan c adalah kelas. Selain itu A adalah jumlah dokumen c yang mengandung t, B adalah jumlah bukan dokumen c yang mengandung t, C adalah jumlah dokumen c yang tidak mengandung t, dan D adalah jumlah bukan dokumen c yang tidak mengandung t.

$$X^2(t,c) = \frac{N(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)}$$

$$= \frac{6((2x4)-(0x0))^2}{(2+0)(0+4)(2+0)(0+4)}$$

$$= \frac{6x64}{2x4x2x4}$$

$$= \frac{384}{64}$$

$$= 6$$

Berikut adalah hasil perhitungan dari nilai *Chi-Square* seluruh term pada data *training* yang dapat dilihat pada Tabel 4.14 hingga Tabel 4.17.

Tabel 4.14 Hasil *Chi-Square* Data Latih Kelas Dishub

No	Term	D1	D2	D3	D4	D5	D6	Dishub				
								A	B	C	D	Chi
1	pasar	1	0	0	0	0	0	1	0	1	4	2,4
2	madyopuro	1	0	0	0	0	0	1	0	1	4	2,4
3	karcis	1	1	0	0	0	0	2	0	0	4	6
4	parkir	1	1	0	0	0	0	2	0	0	4	6
5	tukang	0	1	0	0	0	0	1	0	1	4	2,4
6	Beri	0	1	0	0	0	0	1	0	1	4	2,4
7	mohon	0	0	1	0	1	0	0	2	2	2	1,5
8	tindak	0	0	1	0	0	0	0	1	2	3	0,6
9	wilayah	0	0	1	0	0	0	0	1	2	3	0,6
10	Lurah	0	0	1	0	0	0	0	1	2	3	0,6
11	kauman	0	0	1	0	0	0	0	1	2	3	0,6
12	tugas	0	0	1	0	0	0	0	1	2	3	0,6
13	buang	0	0	1	0	0	0	0	1	2	3	0,6
14	sampah	0	0	1	1	0	0	0	2	2	3	1,5
15	sungai	0	0	1	0	0	0	0	1	2	3	0,6
16	jembatan	0	0	0	1	0	0	0	1	2	3	0,6

No	Term	D1	D2	D3	D4	D5	D6	Dishub				
								A	B	C	D	Chi
17	muharto	0	0	0	1	0	0	0	1	2	3	0,6
18	Tps	0	0	0	1	0	0	0	1	2	3	0,6
19	Baik	0	0	0	0	1	1	0	2	2	2	1,5
20	akses	0	0	0	0	1	0	0	1	2	3	0,6
21	Jalan	0	0	0	0	1	0	0	1	2	3	0,6
22	publik	0	0	0	0	1	0	0	1	2	3	0,6
23	rumah	0	0	0	0	1	0	0	1	2	3	0,6
24	sukun	0	0	0	0	1	0	0	1	2	3	0,6
25	pondok	0	0	0	0	1	0	0	1	2	3	0,6
26	indah	0	0	0	0	1	0	0	1	2	3	0,6
27	dinoyo	0	0	0	0	0	1	0	1	2	3	0,6
28	aspal	0	0	0	0	0	1	0	1	2	3	0,6
29	rusak	0	0	0	0	0	1	0	1	2	3	0,6
30	harap	0	0	0	0	0	1	0	1	2	3	0,6

Pada tabel di atas menjabarkan hasil perhitungan nilai A, B, C, dan D kelas Dishub. Selain itu terdapat nilai *Chi-Square* kelas Dishub pada kolom chi.

Tabel 4.15 Hasil *Chi-Square* Data Latih Kelas DKP

No	Term	D1	D2	D3	D4	D5	D6	DKP				
								A	B	C	D	Chi
1	pasar	1	0	0	0	0	0	0	1	2	3	0,6
2	madyopuro	1	0	0	0	0	0	0	1	2	3	0,6
3	karcis	1	1	0	0	0	0	0	2	2	2	1,5
4	parkir	1	1	0	0	0	0	0	2	2	2	1,5
5	tukang	0	1	0	0	0	0	0	1	2	3	0,6
6	Beri	0	1	0	0	0	0	0	1	2	3	0,6
7	mohon	0	0	1	0	1	0	1	1	1	3	0,375
8	tindak	0	0	1	0	0	0	1	1	1	4	2,4
9	wilayah	0	0	1	0	0	0	1	1	1	4	2,4
10	Lurah	0	0	1	0	0	0	1	1	1	4	2,4
11	kauman	0	0	1	0	0	0	1	1	1	4	2,4

No	Term	D1	D2	D3	D4	D5	D6	DKP				
								A	B	C	D	Chi
12	tugas	0	0	1	0	0	0	1	1	1	4	2,4
13	buang	0	0	1	0	0	0	1	1	1	4	2,4
14	sampah	0	0	1	1	0	0	2	2	0	4	6
15	sungai	0	0	1	0	0	0	1	1	1	4	2,4
16	jembatan	0	0	0	1	0	0	1	1	1	4	2,4
17	muharto	0	0	0	1	0	0	1	1	1	4	2,4
18	Tps	0	0	0	1	0	0	1	1	1	4	2,4
19	Baik	0	0	0	0	1	1	0	0	2	2	1,5
20	akses	0	0	0	0	1	0	0	0	2	3	0,6
21	Jalan	0	0	0	0	1	0	0	0	2	3	0,6
22	publik	0	0	0	0	1	0	0	0	2	3	0,6
23	rumah	0	0	0	0	1	0	0	0	2	3	0,6
24	sukun	0	0	0	0	1	0	0	0	2	3	0,6
25	pondok	0	0	0	0	1	0	0	0	2	3	0,6
26	indah	0	0	0	0	1	0	0	0	2	3	0,6
27	dinoyo	0	0	0	0	0	1	0	0	2	3	0,6
28	aspal	0	0	0	0	0	1	0	0	2	3	0,6
29	rusak	0	0	0	0	0	1	0	0	2	3	0,6
30	harap	0	0	0	0	0	1	0	0	2	3	0,6

Pada tabel di atas menjabarkan hasil perhitungan nilai A, B, C, dan D kelas DKP. Selain itu terdapat nilai *Chi-Square* kelas DKP pada kolom chi.

Tabel 4.16 Hasil *Chi-Square* Data Latih Kelas DPUPPB

No	Term	D1	D2	D3	D4	D5	D6	DPUPPB				
								A	B	C	D	Chi
1	pasar	1	0	0	0	0	0	0	1	2	3	0,6
2	madyopuro	1	0	0	0	0	0	0	1	2	3	0,6
3	karcis	1	1	0	0	0	0	0	2	2	2	1,5
4	parkir	1	1	0	0	0	0	0	2	2	2	1,5
5	tukang	0	1	0	0	0	0	0	1	2	3	0,6
6	Beri	0	1	0	0	0	0	0	1	2	3	0,6

No	Term	D1	D2	D3	D4	D5	D6	DPUPPB				
								A	B	C	D	Chi
7	mohon	0	0	1	0	1	0	1	1	1	3	0,375
8	tindak	0	0	1	0	0	0	0	1	2	3	0,6
9	wilayah	0	0	1	0	0	0	0	1	2	3	0,6
10	Lurah	0	0	1	0	0	0	0	1	2	3	0,6
11	kauman	0	0	1	0	0	0	0	1	2	3	0,6
12	tugas	0	0	1	0	0	0	0	1	2	3	0,6
13	buang	0	0	1	0	0	0	0	1	2	3	0,6
14	sampah	0	0	1	1	0	0	0	2	2	2	1,5
15	sungai	0	0	1	0	0	0	0	1	2	3	0,6
16	jembatan	0	0	0	1	0	0	0	1	2	3	0,6
17	muharto	0	0	0	1	0	0	0	1	2	3	0,6
18	Tps	0	0	0	1	0	0	0	1	2	3	0,6
19	Baik	0	0	0	0	1	1	2	0	0	4	6
20	akses	0	0	0	0	1	0	1	0	1	4	2,4
21	Jalan	0	0	0	0	1	0	1	0	1	4	2,4
22	publik	0	0	0	0	1	0	1	0	1	4	2,4
23	rumah	0	0	0	0	1	0	1	0	1	4	2,4
24	sukun	0	0	0	0	1	0	1	0	1	4	2,4
25	pondok	0	0	0	0	1	0	1	0	1	4	2,4
26	indah	0	0	0	0	1	0	1	0	1	4	2,4
27	dinoyo	0	0	0	0	0	1	1	0	1	4	2,4
28	aspal	0	0	0	0	0	1	1	0	1	4	2,4
29	rusak	0	0	0	0	0	1	1	0	1	4	2,4
30	harap	0	0	0	0	0	1	1	0	1	4	2,4

Pada tabel di atas menjabarkan hasil perhitungan nilai A, B, C, dan D kelas DPUPPB. Selain itu terdapat nilai *Chi-Square* kelas DPUPPB pada kolom chi. Selanjutnya dari hasil nilai *Chi-square* setiap kelas akan dicari nilai rata-rata untuk mengetahui nilai *Chi-square* sesungguhnya.

Tabel 4.17 Hasil *Chi-Square* Data Latih

No	Term	*Chi-Square* Dishub	*Chi-Square* DKP	*Chi-Square* DPUPPB	Rata-rata *Chi-Square*
1	pasar	2,4	0,6	0,6	1,2
2	madyopuro	2,4	0,6	0,6	1,2
3	karcis	6	1,5	1,5	3
4	parkir	6	1,5	1,5	3
5	tukang	2,4	0,6	0,6	1,2
6	Beri	2,4	0,6	0,6	1,2
7	mohon	1,5	0,375	0,375	0,75
8	tindak	0,6	2,4	0,6	1,2
9	wilayah	0,6	2,4	0,6	1,2
10	Lurah	0,6	2,4	0,6	1,2
11	kauman	0,6	2,4	0,6	1,2
12	tugas	0,6	2,4	0,6	1,2
13	buang	0,6	2,4	0,6	1,2
14	sampah	1,5	6	1,5	3
15	sungai	0,6	2,4	0,6	1,2
16	jembatan	0,6	2,4	0,6	1,2
17	muharto	0,6	2,4	0,6	1,2
18	Tps	0,6	2,4	0,6	1,2
19	Baik	1,5	1,5	6	3
20	akses	0,6	0,6	2,4	1,2
21	Jalan	0,6	0,6	2,4	1,2
22	publik	0,6	0,6	2,4	1,2
23	rumah	0,6	0,6	2,4	1,2
24	sukun	0,6	0,6	2,4	1,2
25	pondok	0,6	0,6	2,4	1,2
26	indah	0,6	0,6	2,4	1,2
27	dinoyo	0,6	0,6	2,4	1,2
28	aspal	0,6	0,6	2,4	1,2
29	rusak	0,6	0,6	2,4	1,2

No	Term	*Chi-Square* Dishub	*Chi-Square* DKP	*Chi-Square* DPUPPB	Rata-rata *Chi-Square*
30	harap	0,6	0,6	2,4	1,2

4.2.2.4 Pengurutan Term Hasil *Chi-Square*

Setelah mendapatkan nilai *Chi-square* setiap term, selanjutnya term tersebut akan diurutkan dari term yang memiliki nilai *Chi-Square* tertinggi hingga terendah. Untuk contoh ini akan mengambil fitur sebanyak 75% dari hasil *Chi-square* yaitu 23 term dari 30 term. Pengurutan term dan fitur yang diambil dapat dilihat pada Tabel 4.18.

Tabel 4.18 Hasil Pengurutan Term dari *Chi-Square*

No	Term	*Chi-Square* Dishub	*Chi-Square* DKP	*Chi-Square* DPUPPB	Rata-rata *Chi-Square*
1	karcis	6	1,5	1,5	3
2	parkir	6	1,5	1,5	3
3	sampah	1,5	6	1,5	3
4	Baik	1,5	1,5	6	3
5	pasar	2,4	0,6	0,6	1,2
6	madyopuro	2,4	0,6	0,6	1,2
7	tukang	2,4	0,6	0,6	1,2
8	Beri	2,4	0,6	0,6	1,2
9	tindak	0,6	2,4	0,6	1,2
10	wilayah	0,6	2,4	0,6	1,2
11	Lurah	0,6	2,4	0,6	1,2
12	kauman	0,6	2,4	0,6	1,2
13	tugas	0,6	2,4	0,6	1,2
14	buang	0,6	2,4	0,6	1,2
15	sungai	0,6	2,4	0,6	1,2
16	jembatan	0,6	2,4	0,6	1,2
17	muharto	0,6	2,4	0,6	1,2
18	Tps	0,6	2,4	0,6	1,2
19	akses	0,6	0,6	2,4	1,2
20	Jalan	0,6	0,6	2,4	1,2

No	Term	*Chi-Square* Dishub	*Chi-Square* DKP	*Chi-Square* DPUPPB	Rata-rata *Chi-Square*
21	publik	0,6	0,6	2,4	1,2
22	rumah	0,6	0,6	2,4	1,2
23	sukun	0,6	0,6	2,4	1,2
24	pondok	0,6	0,6	2,4	1,2
25	indah	0,6	0,6	2,4	1,2
26	dinoyo	0,6	0,6	2,4	1,2
27	aspal	0,6	0,6	2,4	1,2
28	rusak	0,6	0,6	2,4	1,2
29	harap	0,6	0,6	2,4	1,2
30	mohon	1,5	0,375	0,375	0,75

Setelah term diurutkan maka diambil sesuai persentasi jumlah fitur yang akan diambil. Pada contoh kali ini mengektraksi fitur sebanyak 75% dari 30 fitur yaitu 23 fitur. Untuk term yang terpilih dari *Chi-Square* dapat dilihat pada Tabel 4.19.

Tabel 4.19 Term Terpilih dari *Chi-Square*

No.	Term
1	karcis
2	parkir
3	sampah
4	baik
5	pasar
6	madyopuro
7	tukang
8	beri
9	tindak
10	wilayah
11	lurah
12	kauman
13	tugas
14	buang

No.	Term
15	sungai
16	jembatan
17	muharto
18	tps
19	akses
20	jalan
21	publik
22	rumah
23	sukun

4.2.2.5 Kombinasi Seleksi Fitur

Pada penelitian kali ini melakukan kombinasi dari kedua seleksi fitur yang digunakan yaitu *Information Gain* dan *Chi-Square.* Untuk kombinasinya menggunakan operasi AND dan OR. Proses kombinasi dilihat dari keberadaan suatu term dari hasil seleksi fitur keduanya. Jika memenuhi syarat operasi AND atau OR maka term akan disimpan. Jika tidak memenuhi maka term akan dibuang. Untuk kombinasi menggunakan operasi AND dapat dilihat pada Tabel 4.20.

Tabel 4.20 Hasil Kombinasi Term Menggunakan Operasi AND

No.	Hasil *Information Gain*	Hasil *Chi-Square*	Keterangan Hasil AND
1	karcis	karcis	Term disimpan
2	parkir	parkir	Term disimpan
3	sampah	sampah	Term disimpan
4	baik	baik	Term disimpan
5	pasar	pasar	Term disimpan
6	madyopuro	madyopuro	Term disimpan
7	tukang	tukang	Term disimpan
8	beri	beri	Term disimpan
9	tindak	tindak	Term disimpan
10	wilayah	wilayah	Term disimpan
11	lurah	lurah	Term disimpan
12	kauman	kauman	Term disimpan
13	tugas	tugas	Term disimpan

No.	Hasil *Information Gain*	Hasil *Chi-Square*	Keterangan Hasil AND
14	buang	buang	Term disimpan
15	sungai	sungai	Term disimpan
16	jembatan	jembatan	Term disimpan
17	muharto	muharto	Term disimpan
18	tps	tps	Term disimpan
19	akses	akses	Term disimpan
20	jalan	jalan	Term disimpan
21	publik	publik	Term disimpan
22	rumah	rumah	Term disimpan
23	sukun	sukun	Term disimpan

Pada kombinasi menggunakan operasi AND akan mengecek term dari kedua hasil seleksi fitur. Apabila suatu term ada pada hasil *Information Gain* dan ada juga pada hasil *Chi-Square* maka term tersebut disimpan. Apabila terdapat term yang hanya ada pada salah satu hasil seleksi fitur maka kata tersebut akan dibuang atau tidak disimpan. Selanjutnya akan menggunakan kombinasi dengan operasi OR. Untuk kombinasi menggunakan operasi OR dapat dilihat pada Tabel 4.21.

Tabel 4.21 Hasil Kombinasi Term Menggunakan Operasi OR

No.	Hasil *Information Gain*	Hasil *Chi-Square*	Keterangan Hasil OR
1	karcis	karcis	Term disimpan
2	parkir	parkir	Term disimpan
3	sampah	sampah	Term disimpan
4	baik	baik	Term disimpan
5	pasar	pasar	Term disimpan
6	madyopuro	madyopuro	Term disimpan
7	tukang	tukang	Term disimpan
8	beri	beri	Term disimpan
9	tindak	tindak	Term disimpan
10	wilayah	wilayah	Term disimpan
11	lurah	lurah	Term disimpan
12	kauman	kauman	Term disimpan

No.	Hasil *Information Gain*	Hasil *Chi-Square*	Keterangan Hasil OR
13	tugas	tugas	Term disimpan
14	buang	buang	Term disimpan
15	sungai	sungai	Term disimpan
16	jembatan	jembatan	Term disimpan
17	muharto	muharto	Term disimpan
18	tps	tps	Term disimpan
19	akses	akses	Term disimpan
20	jalan	jalan	Term disimpan
21	publik	publik	Term disimpan
22	rumah	rumah	Term disimpan
23	sukun	sukun	Term disimpan

Pada kombinasi menggunakan operasi OR akan mengecek term dari kedua hasil seleksi fitur. Apabila suatu term ada pada hasil *Information Gain* dan ada juga pada hasil *Chi-Square* maka term tersebut disimpan. Apabila terdapat term yang hanya ada pada salah satu hasil seleksi fitur maka kata tersebut akan disimpan juga. Dari hasil kombinasi menggunakan operasi AND dan OR akan digunakan selanjutnya pada tahapan klasifikasi. Untuk term hasil kombinasi seleksi fitur dapat dilihat pada Tabel 4.22.

Tabel 4.22 Term Hasil Kombinasi Seleksi Fitur

No.	Kombinasi Menggunakan Operasi AND	Kombinasi Menggunakan Operasi OR
1	karcis	karcis
2	parkir	parkir
3	sampah	sampah
4	baik	baik
5	pasar	pasar
6	madyopuro	madyopuro
7	tukang	tukang
8	beri	beri
9	tindak	tindak
10	wilayah	wilayah
11	lurah	lurah

No.	Kombinasi Menggunakan Operasi AND	Kombinasi Menggunakan Operasi OR
12	kauman	kauman
13	tugas	tugas
14	buang	buang
15	sungai	sungai
16	jembatan	jembatan
17	muharto	muharto
18	tps	tps
19	akses	akses
20	jalan	jalan
21	publik	publik
22	rumah	rumah
23	sukun	sukun

4.2.3 Klasifikasi Teks

Klasifikasi teks adalah proses melabeli kelas atau menentukan kelas suatu dokumen uji berdasarkan data latih yang sudah terlabeli. Metode klasifikasi yang digunakan pada penelitian kali ini adalah *Naïve Bayes*. Metode *Naïve Bayes* merupakan metode dimana akan menghitung semua peluang tiap term terhadap kelas yang ada. Peluang dengan nilai terbesar akan terpilih sebagai kelas pada data uji tersebut. Pada penelitian ini terdapat 3 kelas sesuai dengan SKPD yang ada yaitu Dishub, DKP, dan DPUPPB.

Pada perhitungan untuk klasifikasi dilakukan dua kali karena kumpulan term dari hasil kombinasi seleksi fitur *Information Gain* dan *Chi-Square* berbeda. Dengan tujuan mengetahui hasil klasifikasi masing-masing agar pada pengujian dapat diketahui kombinasi seleksi fitur yang mana lebih baik.

4.2.3.1 Menghitung Kemunculan Setiap Term Pada Dokumen

Sebelum melakukan klasifikasi, hal yang perlu dilakukan adalah menghitung kemunculan setiap term yang ada pada dokumen data latih maupun data uji. Perhitungan ini dilakukan untuk mengetahui berapa banyak term tersebut pada setiap dokumen data latih dan data uji sehingga memudahkan dalam melakukan perhitungan *Naïve Bayes*. Pada contoh kali ini Dokumen 7 merupakan data uji yang digunakan. Term pada dokumen data uji yang tidak ada pada dokumen data latih akan diabaikan atau dibuang. Perhitungan kemunculan term pada setiap dokumen dapat dilihat pada Tabel 4.23.

Tabel 4.23 Perhitungan Kemunculan Setiap Term Pada Dokumen

No.	Term	Dok1	Dok2	Dok3	Dok4	Dok5	Dok6	Dok7
1	karcis	1	1	0	0	0	0	1
2	parkir	1	1	0	0	0	0	1
3	sampah	0	0	1	1	0	0	0
4	baik	0	0	0	0	1	1	0
5	pasar	1	0	0	0	0	0	0
6	madyopuro	1	0	0	0	0	0	0
7	tukang	0	1	0	0	0	0	0
8	beri	0	1	0	0	0	0	0
9	tindak	0	0	1	0	0	0	0
10	wilayah	0	0	1	0	0	0	0
11	lurah	0	0	1	0	0	0	0
12	kauman	0	0	1	0	0	0	0
13	tugas	0	0	1	0	0	0	0
14	buang	0	0	1	0	0	0	0
15	sungai	0	0	1	0	0	0	0
16	jembatan	0	0	0	1	0	0	0
17	muharto	0	0	0	1	0	0	0
18	tps	0	0	0	1	0	0	0
19	akses	0	0	0	0	1	0	0
20	jalan	0	0	0	0	1	0	1
21	publik	0	0	0	0	1	0	0
22	rumah	0	0	0	0	1	0	0
23	sukun	0	0	0	0	1	0	0

4.2.3.2 Menghitung *Prior*

Perhitungan *prior* merupakan perhitungan untuk mengetahui peluang setiap kelas yang ada. Pada data penelitian kali ini terdapat 3 kelas yaitu DIshub, DKP, dan DPUPPB. Perhitungan peluang dengan jumlah dokumen pada kelas tertentu dibagi dengan jumlah dokumen keseluruhan. Berikut contoh perhitungan *Prior* untuk kelas Dishub.

$$P(Dishub) = \frac{jumlah\ dok\ pada\ kelas\ c}{jumlah\ seluruh\ dok}$$
$$= \frac{2}{6}$$
$$= 0{,}333333$$

Perhitungan prior dilakukan ke untuk semua kelas yang ada. Hasil perhitungan prior semua kelas dapat dilihat pada Tabel 4.24.

Tabel 2.24 Hasil Perhitungan *Prior*

No.	Kelas	*Prior*
1	Dishub	0,333333
2	DKP	0,333333
3	DPUPPB	0,333333

4.2.3.3 Perhitungan *Likelihood*

Setelah menghitung nilai *prior*, selanjutnya perhitungan *likelihood*. Perhitungan *likelihood* menghitung probabilitas peluang suatu term dengan syarat kelas tertentu. Berikut salah satu contoh perhitungan *likelihood* pada term "parkir" untuk kelas Dishub dimana |v| adalah jumlah term unik pada data latih, t adalah term, dan c adalah kelas.

$$P(parkir|Dishub) = \frac{count(t,c)+1}{\sum_{w \in V} count(t,c) + |V|}$$
$$= \frac{2+1}{8+23}$$
$$= \frac{3}{31}$$
$$= 0{,}096774$$

Perhitungan *likelihood* dilakukan ke untuk semua term pada semua kelas yang ada. Hasil perhitungan *likelihood* secara keseluruhan dapat dilihat pada Tabel 4.25.

Tabel 4.25 Hasil Perhitungan *Likelihood*

No.	Term	P(t|Dishub)	P(t|DKP)	P(t|DPUPPB)
1	karcis	0,09677419	0,028571429	0,033333333
2	parkir	0,09677419	0,028571429	0,033333333
3	sampah	0,03225806	0,085714286	0,033333333
4	baik	0,03225806	0,028571429	0,1
5	pasar	0,06451613	0,028571429	0,033333333

No.	Term	P(t|Dishub)	P(t|DKP)	P(t|DPUPPB)
6	madyopuro	0,06451613	0,028571429	0,033333333
7	tukang	0,06451613	0,028571429	0,033333333
8	beri	0,06451613	0,028571429	0,033333333
9	tindak	0,03225806	0,057142857	0,033333333
10	wilayah	0,03225806	0,057142857	0,033333333
11	lurah	0,03225806	0,057142857	0,033333333
12	kauman	0,03225806	0,057142857	0,033333333
13	tugas	0,03225806	0,057142857	0,033333333
14	buang	0,03225806	0,057142857	0,033333333
15	sungai	0,03225806	0,057142857	0,033333333
16	jembatan	0,03225806	0,057142857	0,033333333
17	muharto	0,03225806	0,057142857	0,033333333
18	tps	0,03225806	0,057142857	0,033333333
19	akses	0,03225806	0,028571429	0,066666667
20	jalan	0,03225806	0,028571429	0,066666667
21	Publik	0,03225806	0,028571429	0,066666667
22	Rumah	0,03225806	0,028571429	0,066666667
23	Sukun	0,03225806	0,028571429	0,066666667

4.2.3.4 Perhitungan *Posterior*

Perhitungan *posterior* merupakan perhitungan terakhir yang dilakukan pada metode *Naïve Bayes*. Pada tahapan ini sudah memasuki tahapan *testing* dimana telah melibatkan data uji. *Posterior* akan mengalikan nilai *prior* dengan semua nilai *likelihood* pada kelas tertentu yang nilai *likelihood* nya telah dipangkatkan terlebih dahulu sesuai jumlah kemunculan term tersebut pada dokumen uji. Hasil dari perhitungan *posterior* akan menjadi nilai pembandung untuk menentukan kelas yang terpilih. Berikut adalah contoh perhitungan *posterior* pada data uji untuk kelas Dishub.

$$P(Dishub|t) = P(t)\prod P(t|Dishub)$$

$$= P(t)xP(karcis|Dishub)^1xP(parkir|Dishub)^2xP(jalan|Dishub)^1$$

$$= 0{,}333333\ x\ 0{,}09677419\ x\ 0{,}09677419\ x\ 0{,}03225806$$

$$= 0{,}000100702$$

Perhitungan *posterior* dilakukan ke untuk semua kelas yang ada. Hasil perhitungan *posterior* semua kelas dapat dilihat pada Tabel 4.26.

Tabel 4.26 Hasil Perhitungan *Posterior*

No.	Kelas	*Posterior*
1	Dishub	0,000100702
2	DKP	0,000007144
3	DPUPPB	0,000013161

4.2.3.5 Penentuan Kelas

Proses klasifikasi bertujuan untuk melabeli atau mengklasifikasikan suata data pada kelas yang tertentu. Sehingga hasil akhir yang didapatkan pada klasifikasi adalah kelas yang terpilih untuk data tersebut. Pada klasifikasi menggunakan metode *Naïve Bayes* untuk penentuan kelasnya dapat dilihat dari hasil *posterior* setiap kelasnya. Kelas yang memiliki nilai *posterior* paling tinggi, maka data yang diklasifikasikan akan masuk ke dalam kelas tersebut. Berikut tabel hasil kelas yang terpilih untuk dokumen data uji yang diurutkan dari terbesar ke terkecil berdasarkan nilai *posterior* setiap kelas.

Tabel 4.27 Kelas Terpilih

No.	Kelas	*Posterior*
1	Dishub	0,000100702
2	DPUPPB	0,000013161
3	DKP	0,000007144

Pada Tabel 4.27 dapat dilihat kelas yang memiliki *nilai posterior* paling tinggi adalah kelas Dishub. Sehingga hasil klasifikasi menggunakan metode *Naïve Bayes* adalah dokumen uji masuk ke dalam kelas Dishub dengan nilai *posterior* 0,000100702.

4.3 Perancangan Pengujian

Pengujian dilakukan bertujuan untuk mengetahui bagaimana kinerja sistem dari hasil implementasi. Pada penelitian kali ini akan dilakukan pengujian dengan empat skenario yang berbeda. Skenario pengujian yang dilakukan dapat dilihat sebagai berikut.

1. Skenario pertama adalah melakukan klasifikasi tanpa melalui tahapan seleksi fitur.
2. Skenario kedua melakukan klasifikasi dengan melalui tahapan seleski fitur *Information Gain* dan *Chi-Square.* Kedua seleksi fitur tersebut tidak saling berpengaruh karena dilakukan sendiri-sendiri. Hal ini bertujuan untuk mengetahui pengaruh seleksi fitur dalam pengklasifikasian menggunakan *Naïve Bayes* dengan membandingkan hasil dari tanpa melalui tahapan seleksi

fitur dengan hasil yang melalui tahapan seleksi fitur *Information Gain* dan *Chi-Square*.

3. Skenario ketiga adalah pengujian dengan melakukan klasifikasi yang melalui tahapan kombinasi seleksi fitur antara *Information Gain* dan *Chi-Square*. Pada pengujian tersebut dilakukan dua jenis kombinasi yaitu dengan menggunakan operasi AND dan OR. Skenario pengujian ketiga untuk mengetahui pengaruh kombinasi pada seleksi fitur dalam pengklasifikasian menggunakan *Niave Bayes*.
4. Skenario pengujian yang keempat dengan malakukan variasi jumlah fitur yang digunakan untuk tahapan klasifikasi.

Dari keempat skenario di atas akan dilakukan pengujian menggunakan *Confussion Matrix.* Karena terdapat tiga kelas dalam penelitian kali ini, maka pada perhitungan *Confussion Matrix* dilakukan perhitungan sebanyak tiga macam yaitu antara kelas Dishub dengan bukan kelas Dishub, kelas DKP dengan bukan kelas DKP, dan kelas DPUPPB dengan bukan kelas DPUPPB. Hasil *accuracy, precision, recall, dan f-measure* sesungguhnya didapatkan dari mencari nilai rata-rata dari ketiga hasil *confussion matrix* sebelumnya. Perancangan pengujian dapat dilihat pada Tabel 4.28 hingga Tabel 4.29.

Tabel 4.28 Perancangan *Confussion Matrix*

Confussion Matrix Kelas X dan Bukan kelas X			
		Hasil Sebenarnya	
		X	Bukan X
Hasil Sistem	X		
	Bukan X		
Accuracy			
Precision			
Recall			
F-Measure			

Pada tabel di atas, angka "X" merepresentasikan kelas. Setelah mendapat hasil pengujian setiap kelas, maka dicari nilai rata-rata untuk mendapatkan hasil pengujian sesungguhnya.

Tabel 4.29 Perancangan Hasil Pengujian

Accuracy	
Precision	
Recall	
F-Measure	

BAB 5 IMPLEMENTASI

Pada bab ini akan memaparkan hasil implementasi yang sudah didefinisikan sebelumnya di tahap perancangan. Dalam bab ini akan menampilkan *code* program dari hasil implementasi sistem klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* menggunakan metode *Naïve Bayes* dan Kombinasi Seleksi Fitur. Selain itu akan dijelaskan tentang spesifikasi sistem yang digunakan dalam pengimplementasian dan batasan implementasi yang dilakukan.

5.1 Spesifikasi Sistem

Pada penerapan implementasi sistem, dilakukan penerapan sesuai dengan yang didefinisikan pada Bab 3 yaitu Peralatan Pendukung. Pada spesifikasi sistem, akan dipaparkan tentang batasan implementasi pada ruang lingkup perangkat keras dan perangkat lunak.

5.1.1 Spesifikasi Perangkat Keras

Salah satu peralatan pendukung dalam implementasi adalah perangkat keras. Untuk spesifikasi perangkat keras yang digunakan dapat dilihat pada Tabel 5.1

Tabel 5.1 Spesifikasi Perangkat Keras

Nama Komponen	Spesifikasi
Prosesor	Intel® Core™ i3-5200U CPU @ 2.20GHz (4 CPUs)
Memori (RAM)	4GB
Hardisk	HDD 500GB

5.1.2 Spesifikasi Perangkat Lunak

Selain perangkat keras, terdapat perangkat lunak yang menjadi peralatan pendukung dalam melakukan implementasi. Spesifikasi perangkat lunak dapat dilihat pada Tabel 5.2.

Tabel 5.2 Spesfikasi Perangkat Lunak

Nama	Spesfikasi
Sistem Operasi	Windows 10 Pro
Bahasa Pemrograman	Python 3.6
Tools Pemrograman	Spyder
Penyimpanan Data	Microsoft Excel (.csv)

5.2 Batasan Implementasi

Dalam implementasi terdapat batasan-batasan dalam melakukan implementasi. Hal ini bertujuan agar jelas ruang lingkup dari sistem yang dibuat. Untuk batasan implementasi pada sistem klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *Online* menggunakan metode *Naïve Bayes* dan Kombinasi Seleksi Fitur dapat dilihat sebagai berikut.

1. Metode stemming yang digunakan dalam melakukan implementasi adalah Library Sastrawi.
2. Algoritme klasifikasi yang digunakan adalah algoritme *Multinomial Naïve Bayes.*
3. Algoritme seleksi fitur yang digunakan adalah *Chi-Square* dan *Information Gain* yang dikombinasikan menggunakan operasi AND dan OR.
4. Data yang digunakan sebagai data latih maupun data uji berasal dari situs web resmi SAMBAT Online Pemerinta Kota Malang yang dapat diakses di http://sambat.malangkota.go.id/ .
5. Data yang diambil dari tiga kategori atau SKPD yang berbeda yaitu kategori Dishub, DKP, dan DPUPPB.
6. Data yang diambil merupakan dokumen teks berbahasa Indonesia.
7. Data yang diambil disimpan dalam file Microsoft Excel dengan format (.csv)

5.3 Implementasi Algoritme

Pada sistem ini menggunakan algoritme *Naive Bayes* dan Kombinasi Seleksi Fitur dalam pengimplementasiannya. Klasifikasi dokumen SAMBAT *Online* ini terdiri dari lima implementasi algoritme yaitu *text preprocessing, Chi-Square, Information Gain*, Kombinasi Seleksi Fitur, dan *Naïve Bayes*.

5.3.1 Implementasi *Text Preprocessing*

Text Preprocessing melewati beberapa tahapan. Mulai dari *tokenizing*, *filtering/stopwords*, dan *stemming*. Berikut potongan *source code* pada kode program yang dapat dilihat pada Kode Program 5.1 sampai Kode Program 5.3.

5.3.1.1 Implementasi *Stemming*

Stemming merupakan proses mengubah sebuah kata berimbuhan menjadi kata dasar. Implementasi proses *stemming* dapat dilihat pada Kode Program 5.1.

Kode Program 5.1 Implementasi *Stemming*

```
1       with open('file', 'r') as f:
2           reader = f.readlines()

3       factory = StemmerFactory()
4       stemmer = factory.create_stemmer()
5       documents_stemmed = [stemmer.stem(d) for d in
        reader]
```

Penjelasan Kode Program 5.1 :

Baris 1-2 Membaca file sesuai nama file kemudian menyimpan hasilnya pada variabel *reader*.

Baris 3-4 Menggunakan *library stemmer* dan memanggil fungsi *stemming*.

Baris 5 Melakukan *stemming* dari variabel *reader* dan hasil *stemming*-nya disimpan pada variabel *documents_stemmed*.

5.3.1.2 Implementasi *Tokenizing*

Tokenizing merupakan proses memecah kalimat menjadi kata per kata. Selain itu terdapat juga proses *Case Folding* dimana mengubah huruf capital menjadi huruf kecil. Implementasi *tokenizing* dapat dilihat pada Kode Program 5.2.

Kode Program 5.2 Implementasi *Tokenizing*

```
1  tokenize = lambda doc: doc.lower().split(" ")
2  documents_tokenized = [tokenize(d) for d in
   documents_stemmed
```

Penjelasan Kode Program 5.2 :

Baris 1 Memanggil fungsi *doc.lower* untuk mengubah huruf menjadi huruf kecil sekaligus memecah kalimat menggunakan fungsi *split*.

Baris 2 Hasil tokenisasi disimpan ke dalam variabel *documents_tokenizing*.

5.3.1.3 Implementasi *Filtering/Stopwords*

Filtering/Stopwords merupakan proses penghapusan kata yang terdapat dalam *stoplist*. Implementasi *filtering/stopwords* dapat dilihat pada Kode Program 5.3.

Kode Program 5.3 Implementasi *Filtering/Stopwords*

```
1  stop_words=[]
2  with open("stop_words.txt") as f:
3      stopw = f.readlines()
4  stop_words = [x.strip() for x in stopw]

5  filter = lambda doc: [w for w in doc if w not in
   stop_words]
6  documents_filtered = [filter(d) for d in
   documents_tokenized]
```

Penjelasan Kode Program 5.3 :

Baris 1 Membuat *list* dengan nama *stop_words*.

Baris 2-4 Membuka file stop*list* dan menyimpan dalam variabel *stop_words*.

Baris 5 Melakukan *filtering* dengan menyimpan kata yang tidak terdapat dalam *stoplist* yang telah tersimpan dalam variabel *stop_words*.

Baris 6 Menyimpan hasil *filtering* ke dalam variabel *documents_filtered*.

5.3.2 Implementasi *Chi-Square*

Proses seleksi fitur menggunakan algoritme *Chi-Square*, melalui beberapa proses yaitu menghitung nilai A, B, C, dan D kemudian mencari nilai *Chi-Square* masing-masing kelas lalu dirata-ratakan untuk mendapatkan nilai *Chi-Square* sesungguhnya. Setelah mendapat nilai *Chi-Square* maka dilakukan pengurutan term dari terbesar ke terkecil berdasarkan nilai *Chi-Square*. Tahapan terakhir dengan mengekstraksi fitur sesuai dengan *threshold* yang diinputkan. Implementasi *Chi-Square* dapat dilihat pada Kode Program 5.4.

Kode Program 5.4 Implementasi *Chi-Square*

```
        #Input Threshold
1       print("Threshold Fitur yang Digunakan : ")
2       thold = input()

3       dishub_a = []
4       dkp_a = []
5       dpuppb_a = []

6       jum1 = 0
7       jum2 = 0
8       jum3 = 0

        #Hitung Nilai A
9       for i in range(len(unik)):
10          for j in range(len(documents_filtered)):
11                  if label_manual[j]==0:
12                      if unik[i] in
        documents_filtered[j]:
13                          jum1 = jum1 + 1
14                  elif label_manual[j]==1:
15                      if unik[i] in
        documents_filtered[j]:
16                          jum2 = jum2 + 1
17                  elif label_manual[j]==2:
18                      if unik[i] in
        documents_filtered[j]:
19                          jum3 = jum3 + 1
20          dishub_a.append(jum1)
21          jum1 = 0
22          dkp_a.append(jum2)
23          jum2 = 0
24          dpuppb_a.append(jum3)
25          jum3 = 0

        #Hitung Nilai B
26      dishub_b = []
27      dkp_b = []
28      dpuppb_b = []
```

```
        for i in range(len(dishub_a)):
            dishub_b.append(dkp_a[i] + dpuppb_a[i])
            dkp_b.append(dishub_a[i] + dpuppb_a[i])
            dpuppb_b.append(dishub_a[i] + dkp_a[i])

        #Hitung Nilai C
        dishub_c = []
        dkp_c = []
        dpuppb_c = []
        for i in range(len(dishub_a)):
            dishub_c.append(dokDishub - dishub_a[i])
            dkp_c.append(dokDkp - dkp_a[i])
            dpuppb_c.append(dokDpuppb - dpuppb_a[i])

        #Hitung Nilai D
        dishub_d = []
        dkp_d = []
        dpuppb_d = []
        for i in range(len(dishub_a)):
            dishub_d.append((dokDkp + dokDpuppb) -
        dishub_b[i])
            dkp_d.append((dokDishub + dokDpuppb) -
        dkp_b[i])
            dpuppb_d.append((dokDishub + dokDkp) -
        dpuppb_b[i])

        #Hitung Nilai Chi
        dishub_chi = []
        dkp_chi = []
        dpuppb_chi = []
        n = len(documents_filtered)
        for i in range(len(dishub_a)):
            chi1 = (n * (((dishub_a[i] * dishub_d[i])-
        (dishub_c[i] * dishub_b[i])) ** 2)) /
        ((dishub_a[i] + dishub_c[i])*(dishub_b[i] +
        dishub_d[i])*(dishub_a[i] +
        dishub_b[i])*(dishub_c[i] + dishub_d[i]))
            chi2 = (n * (((dkp_a[i] * dkp_d[i])-
        (dkp_c[i] * dkp_b[i])) ** 2)) / ((dkp_a[i] +
        dkp_c[i])*(dkp_b[i] + dkp_d[i])*(dkp_a[i] +
        dkp_b[i])*(dkp_c[i] + dkp_d[i]))
            chi3 = (n * (((dpuppb_a[i] * dpuppb_d[i])-
        (dpuppb_c[i] * dpuppb_b[i])) ** 2)) /
        ((dpuppb_a[i] + dpuppb_c[i])*(dpuppb_b[i] +
        dpuppb_d[i])*(dpuppb_a[i] +
        dpuppb_b[i])*(dpuppb_c[i] + dpuppb_d[i]))
            dishub_chi.append(chi1)
            dkp_chi.append(chi2)
            dpuppb_chi.append(chi3)

        #Hitung Nilai Chi-Square
        avg_chi = []
        for i in range(len(dishub_chi)):
```

```
60              avg_chi.append((dishub_chi[i] + dkp_chi[i] +
        dpuppb_chi[i]) / jumKelas)

        #Membuat list 2 dimensi untuk kata dan nilai
        chi-square masing-masing
61      w, h = 0, len(unik);
62      chisquare = [[0 for x in range(w)] for y in
        range(h)]
63      for i in range(len(unik)):
64          chisquare[i].append(unik[i])
65          chisquare[i].append(avg_chi[i])

        #Mengurutkan Nilai Chi-Square
66      for passnum in range(len(chisquare)-1,0,-1):
67          for i in range(passnum):
68              if chisquare[i][1]<chisquare[i+1][1]:
69                  temp = chisquare[i]
70                  chisquare[i] = chisquare[i+1]
71                  chisquare[i+1] = temp

        #Ekstraksi Fitur
72      tholdchi = len(unik)*(int(thold)/100)
73      jumFiturChi = round(tholdchi,0)

74      x, y = 0, 0;
75      fiturchi = [[0 for x in range(x)] for y in
        range(y)]
76      loop = 0
77      while loop < int(jumFiturChi):
78          fiturchi.append(chisquare[loop])
79          loop = loop + 1;
```

Penjelasan Kode Program 5.4 :

Baris 1 -2 Menginputkan nilai *threshold* untuk ekstrekasi fitur.

Baris 3-5 Membuat *list* untuk menyimpan nilai A dari setiap kelas yang ada.

Baris 6-8 Inisalisasi variabel untuk menghitung nilai A dari setiap kelas yang ada.

Baris 9-25 Membuat perulangan untuk menghitung kemunculan setiap kata unik yang terdapat pada data latih. Terdapat seleksi kondisi untuk mengatahui jumlah kemunculan di kelas tertentu. Jumlah kemunculan tersebut disimpan dalam *list* masing-masing kelas.

Baris 26-28 Membuat *list* untuk menyimpan nilai B dari setiap kelas yang ada.

Baris 29-32 Melakukan perulangan untuk menghitung nilai B dengan menjumlahkan nilai A dari masing-masing kelas selain kelas tertentu. Nilai B kemudian disimpan ke dalam *list* masing-masing kelas.

Baris 33-35 Membuat *list* untuk menyimpan nilai C dari setiap kelas yang ada.

Baris 36-39 Melakukan perulangan untuk menghitung nilai C dengan mencari selisih antara jumlah dokumen dari kelas tertentu dengan nilai A masing-masing dan disimpan ke dalam *list.*

Baris 40-42 Membuat *list* untuk menyimpan nilai D dari setiap kelas yang ada.

Baris 43-46 Melakukan perulangan untuk menghitung nilai D dengan mencari selisi dari jumlah dokumen yang bukan kelas tersebut dengan nilai B masing-masing kemudian disimpan ke dalam *list.*

Baris 47-49 Membuat *list* untuk menyimpan nilai Chi dari setiap kelas yang ada.

Baris 50-57 Melakukan perulangan untuk menghitung nilai chi masing-masing kelas dengan persamaan yang sudah dipaparkan sebelumnya. Kemudian nilai chi tersebut disimpan ke dalam *list* masing-masing kelas.

Baris 58 Membuat *list* untuk menyimpan nilai rata-rata chi dari setiap kelas yang ada.

Baris 59-60 Membuat perulangan untuk menghitung nilai rata-rata dari nilai chi setiap kelas untuk mengetahui nilai *chi-square* yang sesungguhnya.

Baris 61 Inisasliasi nilai w sama dengan 0 dan h sama dengan panjang *list* unik.

Baris 62 Membuat *list* 2 dimensi bernama chisquare dengan baris sesuai nilai w dan kolom sesuai nilai h.

Baris 63-65 Membuat perulangan untuk memasukkan kata unik beserta nilai *chi-square* masing-masing ke dalam *list* dua dimensi.

Baris 66-71 Membuat perulangan untuk mengurutkan nilai *chi-square* dari terbesar ke terkecil.

Baris 72 Menghitung banyaknya fitur yang akan diekstraksi sesuai dengan nilai *threshold* yang diinputkan sebelumnya.

Baris 73 Inisialisasi jumFiturChi dengan membulatkan nilai tholdchi.

Baris 74 Inisialisasi nilai x dan y sama dengan 0.

Baris 76 Inisialisasi nilai loop sama dengan 0.

Baris 77 Melakukan perulangan untuk memasukkan kata hasil ekstraksi fitur ke dalam *list* fiturchi.

5.3.3 Implementasi *Information Gain*

Proses seleksi fitur yang kedua menggunakan algoritme *Informatio Gain.* Terdapat beberapa proses yaitu menghitung peluang kata, peluang kelas, dan peluang kemunculan kata dengan syarat kelas tertentu. Nilai *Information Gain* kemudian diurutkan dan fitur yang terpakai sejumlah batas threshold yang digunakan. Implementasi algoritme *Information Gain* dapat dilihat pada Kode Program 5.5.

Kode Program 5.5 Implementasi *Information Gain*

```
        Hitung IG 1
1       a = dokDishub/len(documents_filtered)
2       kelDishub = a * math.log10(a)

3       b = dokDkp/len(documents_filtered)
4       kelDkp = b * math.log10(b)

5       c = dokDpuppb/len(documents_filtered)
6       kelDpuppb = c * math.log10(c)

7       ig1 = -1*(kelDishub + kelDkp + kelDpuppb)

8       #Hitung Peluang Kata
9       kataDishub = dishub_a
10      kataDkp = dkp_a
11      kataDpuppb = dpuppb_a

12      pelKata = []
        for i in range(len(kataDishub)):
13          kata = (kataDishub[i] + kataDkp[i] +
        kataDpuppb[i])/len(documents_filtered)
14          pelKata.append(kata)

        #Htiung Peluang Dishub
15      pelDishub = []
16      for i in range(len(kataDishub)):
17          a = kataDishub[i] / (kataDishub[i] +
        kataDkp[i] + kataDpuppb[i])
18          if a == 0:
19              dis = 0.0
20          else:
21              dis = a * math.log10(a)
22          pelDishub.append(dis)

        #Htiung Peluang DKP
23      pelDkp = []
24      for i in range(len(kataDkp)):
            b = kataDkp[i] / (kataDishub[i] + kataDkp[i]
        + kataDpuppb[i])
25          if b == 0:
26              dkp = 0.0
27          else:
28              dkp = b * math.log10(b)
29          pelDkp.append(dkp)

        #Htiung Peluang DPUPPB
30      pelDpuppb = []
31      for i in range(len(kataDpuppb)):
32          c = kataDpuppb[i] / (kataDishub[i] +
        kataDkp[i] + kataDpuppb[i])
33          if c == 0:
34              dpuppb = 0.0
```

```
35              else:
36                  dpuppb = c * math.log10(c)
37              pelDpuppb.append(dpuppb)

        #Hitung IG 2
38      ig2 = []
39      for i in range(len(pelDishub)):
            nilai_ig2 = pelKata[i] * (pelDishub[i] +
        pelDkp[i] + pelDpuppb[i])
40          ig2.append(nilai_ig2)

        #Hitung Peluang Negasi Kata
41      pelNegKata = []
42      for i in range(len(pelKata)):
43          negkata = 1-pelKata[i]
44          pelNegKata.append(negkata)

        #Htiung Peluang Dishub Negasi
45      pelDishubNeg = []
46      for i in range(len(kataDishub)):
47          a = (dokDishub-kataDishub[i]) /
        (len(documents_filtered) - (kataDishub[i] +
        kataDkp[i] + kataDpuppb[i]))
48          if a == 0:
49              disNeg = 0.0
50          else:
51              disNeg = a * math.log10(a)
52          pelDishubNeg.append(disNeg)

        #Htiung Peluang DKP Negasi
53      pelDkpNeg = []
54      for i in range(len(kataDkp)):
55          b = (dokDkp-kataDkp[i]) /
        (len(documents_filtered) - (kataDishub[i] +
        kataDkp[i] + kataDpuppb[i]))
56          if b == 0:
57              dkpNeg = 0.0
58          else:
59              dkpNeg = b * math.log10(b)
60          pelDkpNeg.append(dkpNeg)

        #Htiung Peluang DPUPPB Negasi
61      pelDpuppbNeg = []
62      for i in range(len(kataDpuppb)):
63          c = (dokDpuppb-kataDpuppb[i]) /
        (len(documents_filtered) - (kataDishub[i] +
        kataDkp[i] + kataDpuppb[i]))
64          if c == 0:
65              dpuppbNeg = 0.0
66          else:
67              dpuppbNeg = c * math.log10(c)
68          pelDpuppbNeg.append(dpuppbNeg)
```

```
        #Hitung IG 3
69      ig3 = []
70      for i in range(len(pelDishubNeg)):
71          nilai_ig3 = pelNegKata[i] * (pelDishubNeg[i]
        + pelDkpNeg[i] + pelDpuppbNeg[i])
            ig3.append(nilai_ig3)

        #Hitung Nilai Information Gain
72      ig = []
73      for i in range(len(ig2)):
74          nilai_ig = ig1 + ig2[i] + ig3[i]
75          ig.append(nilai_ig)

        #Membuat list 2 dimensi untuk kata dan nilai ig
        masing-masing
76      w, h = 0, len(unik);
77      infoGain = [[0 for x in range(w)] for y in
        range(h)]
78      for i in range(len(unik)):
79          infoGain[i].append(unik[i])
80          infoGain[i].append(ig[i])

        #Mengurutkan Nilai Chi-Square
81      for passnum in range(len(infoGain)-1,0,-1):
82          for i in range(passnum):
83              if infoGain[i][1]<infoGain[i+1][1]:
84                  temp = infoGain[i]
85                  infoGain[i] = infoGain[i+1]
86                  infoGain[i+1] = temp

        #Ekstraksi Fitur
87      tholdig = len(unik)*(int(thold)/100)
88      jumFiturIg = round(tholdig,0)

89      x, y = 0, 0;
90      fiturig = [[0 for x in range(x)] for y in
        range(y)]
91      loop = 0
92      while loop < int(jumFiturIg):
93          fiturig.append(infoGain[loop])
94          loop = loop + 1;
```

Penjelasan Kode Program 5.5 :

Baris 1-2 Menghitung peluang kelas Dishub dan dikalikan dengan log dari peluang kelas Dishub tersebut.

Baris 3-4 Menghitung peluang kelas DKP dan dikalikan dengan log dari peluang kelas Dishub tersebut.

Baris 5-6 Menghitung peluang kelas DPUPPB dan dikalikan dengan log dari peluang kelas Dishub tersebut.

Baris 7 Menghitung nilai ig1 dengan menjumlahkan semua peluang kelas dan dikalikan -1.

Baris 9-10 Inisialisasi untuk menyimpan nilai kemunculan kata setiap dokumen.

Baris 12-14 Membuat perulangan untuk menghitung peluang setiap kata kemudian nilai tersebut dimasukkan ke dalam *list* pelKata.

Baris 15-22 Menghitung peluang setiap kata dengan syarat kelas Dishub. Terdapat seleksi kondisi apabila nilai peluangnya 0 maka otomatis nilainya akan 0 apabila bukan 0 maka akan dilakukan perkalian dengan log dari nilai tersebut. Kemudian nilai tersebut dimasuukan ke dalam *list* pelDishub.

Baris 22-29 Menghitung peluang setiap kata dengan syarat kelas DKP. Terdapat seleksi kondisi apabila nilai peluangnya 0 maka otomatis nilainya akan 0 apabila bukan 0 maka akan dilakukan perkalian dengan log dari nilai tersebut. Kemudian nilai tersebut dimasuukan ke dalam *list* pelDkp.

Baris 30-37 Menghitung peluang setiap kata dengan syarat kelas DPUPPB. Terdapat seleksi kondisi apabila nilai peluangnya 0 maka otomatis nilainya akan 0 apabila bukan 0 maka akan dilakukan perkalian dengan log dari nilai tersebut. Kemudian nilai tersebut dimasuukan ke dalam *list* pelDpuppb.

Baris 38-40 Melakukan perulangan untuk menghitung nilai ig2 dengan menjumlahkan semua peluang kata dengan syarat kelas teretntu. Nilai tersebut kemudian dimasukkan ke dalam *list* ig2.

Baris 41-44 Membuat perulangan untuk menghitung peluang setiap negasi kata kemudian nilai tersebut dimasukkan ke dalam *list* pelNegKata.

Baris 45-52 Menghitung peluang setiap negasi kata dengan syarat kelas Dishub. Terdapat seleksi kondisi apabila nilai peluangnya 0 maka otomatis nilainya akan 0 apabila bukan 0 maka akan dilakukan perkalian dengan log dari nilai tersebut. Kemudian nilai tersebut dimasuukan ke dalam *list* pelNegDishub.

Baris 53-60 Menghitung peluang setiap negasi kata dengan syarat kelas DKP. Terdapat seleksi kondisi apabila nilai peluangnya 0 maka otomatis nilainya akan 0 apabila bukan 0 maka akan dilakukan perkalian dengan log dari nilai tersebut. Kemudian nilai tersebut dimasuukan ke dalam *list* pelNegDkp.

Baris 61-68 Menghitung peluang setiap negasi kata dengan syarat kelas DPUPPB. Terdapat seleksi kondisi apabila nilai peluangnya 0 maka otomatis nilainya akan 0 apabila bukan 0 maka akan dilakukan perkalian dengan log dari nilai tersebut. Kemudian nilai tersebut dimasuukan ke dalam *list* pelNegDpuppb.

Baris 69-71 Melakukan perulangan untuk menghitung nilai ig3 dengan menjumlahkan semua peluang negasi kata dengan syarat kelas teretntu. Nilai tersebut kemudian dimasukkan ke dalam *list* ig3.

Baris 72-75 Melakukan perulangan untuk menghitung nilai ig dengan menjumlahkan semua nilai ig1, ig2, dan ig3. Kemudian nilai tersebut dimasukkan ke dalam *list* nilai_ig.

Baris 76-80 Membuat *list* dua dimensi untuk menyimpan setiap kata dan nilai ig-nya.

Baris 81-86 Melakukan perulangan untuk mengurutkan kata dari terbesar ke terkecil.

Baris 87-88 Mengambil kata dengan jumlah sesuai dengan nilai *threshold* yang dimasukkan.

Baris 88-94 Melakukan perulangan untuk menyimpan kata yang akan digunakan pada *list* fiturig.

5.3.4 Kombinasi Seleksi Fitur

Pada implementasi kali ini menggunakan dua seleksi fitur yang dikombinasikan yaitu *Chi-Square* dan *Information Gain*. Proses kombinasi digunakan ada dua yaitu AND dan OR. Sehingga terdapat dua jenis kumpulan fitur yang akan dilakukan perhitungan Naïve Bayes. Untuk implementasi kombinasi seleksi fitur dapat dilihat pada Kode Program 5.6.

Kode Program 5.6 Implementasi Kombinasi Seleksi Fitur

```
1           for i in range(len(fiturchi)):
2               termIg.append(fiturig[i][0])
3               termChi.append(fiturchi[i][0])

                #Operator AND
4           fiturAnd = []
5           for i in range(len(termChi)):
6               if termChi[i] in termIg:
7                   fiturAnd.append(termChi[i])
8               else:
9                   pass

                #Operasi OR
10          fiturOr = termChi
11          for i in range(len(termIg)):
12              if termIg[i] in fiturOr:
13                  pass
14              else:
15                  fiturOr.append(termIg[i])
```

Penjelasan Kode Program 5.6 :

Baris 1-3 Melakukan perulangan untuk menyimpan fitur hasil *Chi-square* dan *Information Gain.*

Baris 4-9 Melakukan perulangan untuk kombinasi menggunakan operasi AND. Terdapat seleksi kondisi jika *list* fitur *Chi-square* terdapat dalam *list* fitur *Information Gain* maka term tersebut akan digunakan. Jika tidak maka tidak digunakan.

Baris 10-15 Melakukan perulangan untuk kombinasi menggunakan operasi OR. Terdapat seleksi kondisi jika *list* fitur *Information Gain* terdapat dalam *list* fitur *Chi-square* maka term tersebut tidak disimpan. Jika tidak maka akan disimpan.

5.3.5 Implementasi *Raw TF*

Sebelum melakukan perhitungan *Naïve Bayes,* diperlukan terlebih dahulu perhitungan *term frequency.* Perhitungan tersebut menghitung kemunculan setiap kata pada dokumen latih. Terdapat dua jenis fitur yang dilakukan *Raw TF* yaitu hasil fitur kombinasi fitur dengan operasi AND dan OR. Untuk Implementasi algoritme *Raw TF* dapat dilihat pada Kode Program 5.7.

Kode Program 5.7 Implementasi *Raw TF*

```
        #Raw TF Fitur AND
1       dishub_tfand = []
2       dkp_tfand = []
3       dpuppb_tfand = []

4       raw1 = 0
5       raw2 = 0
6       raw3 = 0
7       for i in range(len(fiturAnd)):
8           for j in range(len(documents_filtered)):
9                   if label_manual[j]==0:
10                      raw1 = raw1 +
        documents_filtered[j].count(fiturAnd[i])
11                  elif label_manual[j]==1:
12                      raw2 = raw2 +
        documents_filtered[j].count(fiturAnd[i])
13                  elif label_manual[j]==2:
14                      raw3 = raw3 +
        documents_filtered[j].count(fiturAnd[i])
15          dishub_tfand.append(raw1)
16          raw1 = 0
17          dkp_tfand.append(raw2)
18          raw2 = 0
19          dpuppb_tfand.append(raw3)
20          raw3 = 0

        #Raw TF Fitur OR
21      dishub_tfor = []
22      dkp_tfor = []
23      dpuppb_tfor = []
```

```
24          raw1 = 0
25          raw2 = 0
26          raw3 = 0
27          for i in range(len(fiturOr)):
28              for j in range(len(documents_filtered)):
29                      if label_manual[j]==0:
30                          raw1 = raw1 +
        documents_filtered[j].count(fiturOr[i])
31                      elif label_manual[j]==1:
32                          raw2 = raw2 +
        documents_filtered[j].count(fiturOr[i])
33                      elif label_manual[j]==2:
34                          raw3 = raw3 +
        documents_filtered[j].count(fiturOr[i])
35              dishub_tfor.append(raw1)
36              raw1 = 0
37              dkp_tfor.append(raw2)
38              raw2 = 0
39              dpuppb_tfor.append(raw3)
40              raw3 = 0
```

Penjelasan Kode Program 5.7 :

Baris 1-3 Membuat list untuk menyimpan nilai TF fitur AND dari setiap kelas.

Baris 4-6 Inisialisasi beberapa variabel dengan nilai 0

Baris 7-20 Melakukan perulangan untuk menghitung nilai TF fitur AND setiap kelas. Terdapat seleksi kondisi untuk menghitung nilai TF sesuai kelasnya dengan menambahkan variabel raw1, raw2, atau raw3 apabila kata tersebut terdapat dalam dokumen tertentu. Kemudian nilai TF setiap kelas dimasukkan ke dalam *list* masing-masing.

Baris 21-23 Membuat list untuk menyimpan nilai TF fitur OR dari setiap kelas.

Baris 24-26 Inisialisasi beberapa variabel dengan nilai 0

Baris 27-40 Melakukan perulangan untuk menghitung nilai TF fitur OR setiap kelas. Terdapat seleksi kondisi untuk menghitung nilai TF sesuai kelasnya dengan menambahkan variabel raw1, raw2, atau raw3 apabila kata tersebut terdapat dalam dokumen tertentu. Kemudian nilai TF setiap kelas dimasukkan ke dalam *list* masing-masing.

5.3.6 Implementasi *Naïve Bayes*

Setelah mendapatkan nilai TF setiap fitur maka dilakukan tahapan klasifikasi menggunakan algortime *Naïve Bayes.* Proses perhitungan *Naïve Bayes* terdapat juga beberapa proses yaitu, perhitungan *prior, likelihood, posterior,* dan penentuan kelas terpilih. Dari proses-proses tersebut terbagi dua dalam proses *training* dan proses *testing.* Untuk implementasi algoritme *Naïve Bayes* dapat dilihat sebagai berikut.

5.3.6.1 Implementasi Perhitungan *Prior*

Proses pertama dalam *Naïve Bayes* adalah menghitung nilai *prior*. Menghitung nilai *prior* dengan menghitung nilai peluang setiap kelas yang ada. Implementasi perhitungan *prior* dapat dilihat pada Kode Program 5.8.

Kode Program 5.8 Implementasi Perhitungan *Prior*

```
1       priorDishub = dokDishub/len(documents_filtered)
2       priorDkp = dokDkp/len(documents_filtered)
3       priorDpuppb = dokDpuppb/len(documents_filtered)
```

Penjelasan Kode Program 5.8 :

Baris 1 Menghitung prior dari kelas Dishub dengan jumlah dokumen kelas Dishub dibagi dengan jumlah semua dokumen.

Baris 2 Menghitung prior dari kelas DKP dengan jumlah dokumen kelas DKP dibagi dengan jumlah semua dokumen.

Baris 3 Menghitung prior dari kelas DPUPPB dengan jumlah dokumen kelas DPUPPB dibagi dengan jumlah semua dokumen.

5.3.6.2 Implementasi Perhitungan *Likelihood*

Proses selanjutnya menghitung nilai *likelihood* setiap kata. Perhitungan *likelihood* atau *conditional probability* menggunakan algoritme *smoothing*. Implementasi perhitungan *likelihood* dapat dilihat pada Kode Program 5.9.

Kode Program 5.9 Implementasi Perhitungan *Likelihood*

```
        #Likelihood Fitur And
1       UnikAnd = len(fiturAnd)

        #Hitung Likelihood Dishub
2       jumDishub = 0
3       for i in range(len(dishub_tfand)):
4           jumDishub = jumDishub + dishub_tfand[i]

5       likeDishubAnd = []
6       for i in range(len(fiturAnd)):
7           like = (dishub_tfand[i] + 1) / (jumDishub +
        UnikAnd)
8           likeDishubAnd.append(like)

        #Hitung Likelihood DKP
9       jumDkp = 0
10      for i in range(len(dkp_tfand)):
11          jumDkp = jumDkp + dkp_tfand[i]

12      likeDkpAnd = []
13      for i in range(len(fiturAnd)):
14          like = (dkp_tfand[i] + 1) / (jumDkp +
        UnikAnd)
15          likeDkpAnd.append(like)
```

```
        #Hitung Likelihood DPUPPB
16      jumDpuppb = 0
17      for i in range(len(dpuppb_tfand)):
18          jumDpuppb = jumDpuppb + dpuppb_tfand[i]

19      likeDpuppbAnd = []
20      for i in range(len(fiturAnd)):
21          like = (dpuppb_tfand[i] + 1) / (jumDpuppb +
        UnikAnd)
22          likeDpuppbAnd.append(like)

        #Likelihood Fitur OR
23      UnikOr = len(fiturOr)

        #Hitung Likelihood Dishub
24      jumDishub = 0
25      for i in range(len(dishub_tfor)):
26          jumDishub = jumDishub + dishub_tfor[i]

27      likeDishubOr = []
28      for i in range(len(fiturOr)):
29          like = (dishub_tfor[i] + 1) / (jumDishub +
        UnikOr)
30          likeDishubOr.append(like)

        #Hitung Likelihood DKP
31      jumDkp = 0
32      for i in range(len(dkp_tfor)):
33          jumDkp = jumDkp + dkp_tfor[i]

34      likeDkpOr = []
35      for i in range(len(fiturOr)):
36          like = (dkp_tfor[i] + 1) / (jumDkp + UnikOr)
37          likeDkpOr.append(like)

        #Hitung Likelihood DPUPPB
38      jumDpuppb = 0
39      for i in range(len(dpuppb_tfor)):
40          jumDpuppb = jumDpuppb + dpuppb_tfor[i]

41      likeDpuppbOr = []
42      for i in range(len(fiturOr)):
43          like = (dpuppb_tfor[i] + 1) / (jumDpuppb +
        UnikOr)
44          likeDpuppbOr.append(like)
```

Penjelasan Kode Program 5.9 :

Baris 1 Inisialisasi nilai unik fitur AND.

Baris 2-4 Melakukan perulangan untuk menghitung jumlah fitur AND yang terdapat dalam kelas Dishub.

Baris 5-8 Melakukan perulangan untuk menghitung perhitungan *likelihood* fitur AND sesuai dengan persamaannya lalu disimpan ke dalam *list likelihood* fitur AND untuk kelas Dishub.

Baris 9-11 Melakukan perulangan untuk menghitung jumlah fitur AND yang terdapat dalam kelas DKP.

Baris 12-14 Melakukan perulangan untuk menghitung perhitungan *likelihood* fitur AND sesuai dengan persamaannya lalu disimpan ke dalam *list likelihood* fitur AND untuk kelas DKP.

Baris 16-18 Melakukan perulangan untuk menghitung jumlah fitur AND yang terdapat dalam kelas DPUPPB.

Baris 19-22 Melakukan perulangan untuk menghitung perhitungan *likelihood* fitur AND sesuai dengan persamaannya lalu disimpan ke dalam *list likelihood* fitur AND untuk kelas DPUPPB.

Baris 23 Insialisasi nilai unik fitur OR.

Baris 24-26 Melakukan perulangan untuk menghitung jumlah fitur OR yang terdapat dalam kelas Dishub.

Baris 27-30 Melakukan perulangan untuk menghitung perhitungan *likelihood* fitur OR sesuai dengan persamaannya lalu disimpan ke dalam *list likelihood* fitur OR untuk kelas Dishub.

Baris 31-33 Melakukan perulangan untuk menghitung jumlah fitur OR yang terdapat dalam kelas DKP.

Baris 34-37 Melakukan perulangan untuk menghitung perhitungan *likelihood* fitur OR sesuai dengan persamaannya lalu disimpan ke dalam *list likelihood* fitur OR untuk kelas DKP.

Baris 38-40 Melakukan perulangan untuk menghitung jumlah fitur OR yang terdapat dalam kelas DPUPPB.

Baris 41-44 Melakukan perulangan untuk menghitung perhitungan *likelihood* fitur OR sesuai dengan persamaannya lalu disimpan ke dalam *list likelihood* fitur OR untuk kelas DPUPPB.

5.3.6.3 Impementasi Perhitungan *Posterior*

Perhitungan terakhir dalam algoritme Naïve Bayes adalah perhitungan *posterior.* Perhitungan *posterior* melibatkan nilai *prior* dan *likelihood.* Hasil *posterior* ini akan digunakan untuk penentuan kelas yang terpilih. Implementasi perhitungan *posterior* dapat dilihat pada Kode Program 5.10.

Kode Program 5.10 Implementasi Perhitungan *Posterior*

```
1         while loop < batas:
2             tfuji_and = []
3             rawtf = 0
4             for i in range(len(fiturAnd)):
5
```

```
                rawtf = rawtf +
6       documents_testing[loop].count(fiturAnd[i])
7               tfuji_and.append(rawtf)
                rawtf = 0
8
9           pos1 = 0
10          pos2 = 0
            pos3 = 0
11
12          posDishub = []
13          posDkp = []
14          posDpuppb = []
15          for i in range(len(fiturAnd)):
16              pos1 = likeDishubAnd[i] ** tfuji_and[i]
17              posDishub.append(pos1)
18              pos2 = likeDkpAnd[i] ** tfuji_and[i]
19              posDkp.append(pos2)
20              pos3 = likeDpuppbAnd[i] ** tfuji_and[i]
                posDpuppb.append(pos3)
21
22          posteriorDishub = 1
23          posteriorDkp = 1
24          posteriorDpuppb = 1
25          for i in range(len(fiturAnd)):
                posteriorDishub = posteriorDishub *
26      posDishub[i]
27              posteriorDkp =  posteriorDkp * posDkp[i]
28              posteriorDpuppb = posteriorDpuppb *
        posDpuppb[i]
29
            posteriorDishub = posteriorDishub *
30      priorDishub
31          posterior1and.append(posteriorDishub)
32          posteriorDkp = posteriorDkp * priorDkp
33          posterior2and.append(posteriorDkp)
            posteriorDpuppb = posteriorDpuppb *
34      priorDpuppb
            posterior3and.append(posteriorDpuppb)
35
36      tfuji_or = []
37          rawtf = 0
38          for i in range(len(fiturOr)):
                rawtf = rawtf +
39      documents_testing[loop].count(fiturOr[i])
40              tfuji_or.append(rawtf)
                rawtf = 0
41
42          pos1 = 0
43          pos2 = 0
            pos3 = 0
44
45          posDishub = []
46          posDkp = []
```

```
47              posDpuppb = []
48              for i in range(len(fiturOr)):
49                  pos1 = likeDishubOr[i] ** tfuji_or[i]
50                  posDishub.append(pos1)
51                  pos2 = likeDkpOr[i] ** tfuji_or[i]
52                  posDkp.append(pos2)
53                  pos3 = likeDpuppbOr[i] ** tfuji_or[i]
                    posDpuppb.append(pos3)
54
55              posteriorDishub = 1
56              posteriorDkp = 1
57              posteriorDpuppb = 1
58              for i in range(len(fiturOr)):
                    posteriorDishub = posteriorDishub *
59          posDishub[i]
60                  posteriorDkp =  posteriorDkp * posDkp[i]
                    posteriorDpuppb = posteriorDpuppb *
            posDpuppb[i]
61
                    posteriorDishub = posteriorDishub *
62          priorDishub
63              posterior1or.append(posteriorDishub)
64              posteriorDkp = posteriorDkp * priorDkp
65              posterior2or.append(posteriorDkp)
                posteriorDpuppb = posteriorDpuppb *
66          priorDpuppb
                posterior3or.append(posteriorDpuppb)
67
            loop = loop + 1
```

Penjelasan Kode Program 5.10 :

Baris 1 Membuat perulangan while hingga batas panjang data uji.

Baris 2-7 Menghitung nilai TF fitur AND untuk data uji lalu disimpan ke *list* nilai TF fitur AND.

Baris 8-10 Insialisasi beberapa variabel dengan nilai 0.

Baris 11-13 Membuat *list* untuk menyimpan nilai *likelihood* yang sudah dipangkatkan dengan nilai TF fitur AND data uji.

Baris 14-20 Melakukan perulangan untuk menghitung keseluruhan nilai *likelihood* setiap fitur berdasarkan nilai TF fitur AND data uji dengan cara dipangkatkan dan disimpan ke *list* masing-masing kelas.

Baris 21-23 Insialisasi beberapa variabel dengan nilai 1.

Baris 24-28 Melakukan perulangan untuk menghitung nilai *likelihood* keseluruhan setiap kelas dengan mengalikan semua nilai *likelihood* setiap fitur AND.

Baris 29-34 Menghitung nilai *posterior* fitur AND setiap kelas dengan mengalikan nilai *prior* dengan *likelihood* masing-masing kelas. Nilai

posterior tersebut dimasukkan ke dalam *list* nilai *posterior* sesuai kelas.

Baris 35-40 Menghitung nilai TF fitur OR untuk data uji lalu disimpan ke *list* nilai TF fitur OR.

Baris 41-43 Insialisasi beberapa variabel dengan nilai 0.

Baris 44-46 Membuat *list* untuk menyimpan nilai *likelihood* yang sudah dipangkatkan dengan nilai TF fitur OR data uji.

Baris 47-53 Melakukan perulangan untuk menghitung keseluruhan nilai *likelihood* setiap fitur berdasarkan nilai TF fitur OR data uji dengan cara dipangkatkan dan disimpan ke *list* masing-masing kelas.

Baris 54-56 Insialisasi beberapa variabel dengan nilai 1.

Baris 57-60 Melakukan perulangan untuk menghitung nilai *likelihood* keseluruhan setiap kelas dengan mengalikan semua nilai *likelihood* setiap fitur OR.

Baris 61-66 Menghitung nilai *posterior* fitur OR setiap kelas dengan mengalikan nilai *prior* dengan *likelihood* masing-masing kelas. Nilai *posterior* tersebut dimasukkan ke dalam *list* nilai *posterior* sesuai kelas.

Baris 67 Menambahkan nilai loop untuk batas perulangan.

5.3.6.4 Implementasi Penentuan Kelas

Setelah mendapat nilai *posterior,* maka dapat dilakukan penentuan kelas. Penentuan kelas dilakukan dengan mencari nilai *posterior* kelas yang terbesar. Implementasi penentuan kelas dapat dilihat pada Kode Program 5.11.

Kode Program 5.11 Implementasi Penentuan Kelas

```
1       if posteriorDishub > posteriorDkp:
2               if posteriorDishub > posteriorDpuppb :
3                   kelasAnd.append(0)
4               else:
5                   kelasAnd.append(2)
6           elif posteriorDkp > posteriorDishub :
7               if posteriorDkp > posteriorDpuppb :
8                   kelasAnd.append(1)
9               else :
10                  kelasAnd.append(2)

11          if posteriorDishub > posteriorDkp:
12              if posteriorDishub > posteriorDpuppb :
13                  kelasOr.append(0)
14              else:
15                  kelasOr.append(2)
16          elif posteriorDkp > posteriorDishub :
```

```
17                  if posteriorDkp > posteriorDpuppb :
18                      kelasOr.append(1)
19                  else :
20                      kelasOr.append(2)
```

Penjelasan Kode Program 5.11 :

Baris 1-10 Mengecek nilai *posterior* fitur AND dengan melakukan seleksi kondisi untuk mengetahui nilai *posterior* yang tertinggi dan akan dijadikan kelas yang terpilih.

Baris 11-20 Mengecek nilai *posterior* fitur OR dengan melakukan seleksi kondisi untuk mengetahui nilai *posterior* yang tertinggi dan akan dijadikan kelas yang terpilih.

BAB 6 PENGUJIAN DAN ANALISIS

Pada bab ini akan memaparkan hasil dari pengujian setiap skenario yang telah didefinisikan pada bab perancangan bagian perancangan pengujian. Pada bab ini akan menjelaskan skenario pengujiannya dan akan memberikan hasil beserta analisa dari hasil tersebut. Pada pengujian kali ini adalah sebanyak 204 data. Untuk 80% data atau 162 data digunakan untuk data latih dengan pembagian 80 kelas Dishub, 29 kelas DKP, dan 53 kelas DPUPPB. Sedangkan 20% atau 37 data digunakan untuk data uji dengan pembagian 20 kelas Dishub, 8 kelas DKP, dan 14 kelas DPUPPB.

6.1 Skenario Pengujian Tanpa Menggunakan Seleksi Fitur

Skenario pengujian yang pertama adalah melakukan klasifikasi menggunakan metode *Naïve Bayes* tanpa melalui proses seleksi fitur. Skenario ini bertujuan untuk mengetahui bagaimana hasil sistem apabila tidak melalui proses seleksi fitur.

6.1.1 Hasil dan Pembahasan Tanpa Menggunakan Seleksi Fitur

Setelah dilakukan pengujian, didapatkan hasil dari sistem untuk dilakukan analisis. Hasil yang diberikan berupa hasil *accuracy, precision, recall,* dan *f-measure.* Untuk hasil dari pengujian tanpa menggunakan seleksi fitur dapat dilihat pada Tabel 6.1.

Tabel 6.1 Hasil Pengujian Tanpa Menggunakan Seleksi Fitur

Accuracy	80,95%
Precision	73,45%
Recall	76,14%
F-Measure	73,23%

Pada hasil pengujian di atas dalam melakukan klasifikasi tanpa menggunakan seleksi fitur didapatkan hasil *accuracy* sebesar 80,95%, *precision* sebesar 73,45%, *recall* sebesar 76,14%, dan *f-measure* sebesar 73,23%. Dari hasil tersebut akan dibandingkan dengan hasil dari pengujian klasifikasi menggunakan seleksi fitur yaitu *Chi-square* dan *Information Gain.*

6.1.2 Analisis Tanpa Menggunakan Seleksi Fitur

Setelah dilakukan pengujian tanpa menggunakan seleksi fitur, dihasilkan tingkat akurasi sebesar 80,95%. Dimana sistem mampu mengklasifikasikan 34 dokumen uji sesuai dengan kelasnya. Sehingga dari hasil pengujian tanpa menggunakan seleksi fitur, dapat dikatakan metode *Naïve Bayes* mampu memberikan hasil yang baik dalam melakukan klasifikasi dokumen pengaduan SAMBAT *online.*

6.2 Skenario Menggunakan Seleksi Fitur

Skenario kedua yang dilakukan adalah melakukan klasifikasi dengan melewati tahapan seleksi fitur. Untuk seleksi fitur yang digunakan ada dua yaitu, *Chi-square* dan *Information Gain.* Skenario ini bertujuan untuk mengetahui bagaimana hasil sistem apabila melalui tahapan seleksi fitur sebelum melakukan klasifikasi sehingga dapat diketahui pengaruh seleksi fitur dalam melakukan klasifikasi menggunakan metode *Naïve Bayes.*

6.2.1 Menggunakan *Chi-Square*

Seleksi fitur pertama yang digunakan adalah *Chi-square.* Jumlah fitur yang digunakan pada pengujian ini akan dilakukan variasi yaitu, 25%, 50%, 75%, dan 100%. *Threshold* tersebut merupakan jumlah fitur yang akan digunakan. Hasil yang diberikan berupa hasil *accuracy, precision, recall,* dan *f-measure.* Untuk hasil dari pengujian dengan menggunakan seleksi fitur *Chi-square* dapat dilihat pada Tabel 6.2.

Tabel 6.2 Hasil Pengujian Menggunakan Seleksi Fitur *Chi-Square*

	Jumlah Fitur			
	25%	**50%**	**75%**	**100%**
Accuracy	83,33%	83,33%	80,95%	80,95%
Precision	75,12%	75,12%	70,95%	73,45%
Recall	81,14%	81,14%	76,70%	76,14%
F-Measure	75,41%	75,41%	70,05%	73,23%

Pada tabel di atas dapat dilihat hasil pengujian dari skenario pengujian menggunakan seleksi fitur dengan algoritme *Chi-square*. Pada pengujian ini menggunakan jumlah fitur sebesar 25%, 50%, 75%, dan 100% dari keseluruhan fitur yang ada. Hasil pengujian sistem tersebut dapat dilihat variasi *threshold* fitur yang optimal adalah 25% dan 50%. Dimana *threshold* tersebut dihasilkan nilai *accuracy* yang didapatkan adalah 83,33%, *precision* sebesar 75,12%, *recall* sebesar 81,14%, dan *f-measure* sebesar 75,41%. Dengan menggunakan fitur sebesar 25% dan 50%, *Chi-Square* mampu memberikan hasil urutan kata yang relevan di 25% dan 50% kata tersebut. Sebaliknya dengan menggunakan 75% dan 100% fitur membuat sistem menggunakan lebih banyak kata dan keseluruhan kata dimana terdapat kata yang tidak terlalu relevan untuk digunakan. Sehingga pertambahan jumlah fitur pada hasil *Chi-Square* membuat kata yang tidak relevan memungkinkan untuk ikut digunakan dan membuat hasil sistem tidak optimal.

6.2.2 Menggunakan *Information Gain*

Metode seleksi fitur lainnya yang digunakan adalah *Information Gain.* Jumlah fitur yang digunakan pada pengujian kali ini disamakan dengan variasi jumlah fitur yang digunakan pada pengujian menggunakan *Chi-square* yaitu 25%, 50%, 75%,

dan 100%. *Threshold* tersebut merupakan jumlah fitur yang akan digunakan. Hasil yang diberikan berupa hasil *accuracy, precision, recall,* dan *f-measure.* Untuk hasil dari pengujian dengan menggunakan seleksi fitur *Information Gain* dapat dilihat pada Tabel 6.3.

Tabel 6.3 Hasil Pengujian Menggunakan Seleksi Fitur *Information Gain*

	Jumlah Fitur			
	25%	**50%**	**75%**	**100%**
Accuracy	78,57%	83,33%	83,33%	80,95%
Precision	68,57%	75,12%	75,12%	73,45%
Recall	74,94%	81,14%	81,14%	76,14%
F-Measure	68,03&	75,41%	75,41%	73,23%

Pada tabel di atas dapat dilihat hasil pengujian dari skenario pengujian menggunakan seleksi fitur dengan algoritme *Information Gain*. Pada pengujian ini menggunakan variasi jumlah fitur sebesar 25%, 50%, 75%, dan 100% dari keseluruhan fitur yang ada. Hasil pengujian sistem tersebut dihasilkan nilai *threshold* yang paling optimal adalah 50% dan 75% dengan nilai *accuracy* yang didapatkan adalah 83,33%, *precision* sebesar 75,12%, *recall* sebesar 81,14%, dan *f-measure* sebesar 75,41%. Dengan menggunakan fitur sebesar 25%, *Information Gain* belum memberikan hasil urutan kata yang relevan di 25% kata tersebut. Sebaliknya dengan menggunakan 100% fitur membuat sistem menggunakan keseluruhan kata dimana terdapat kata yang tidak terlalu relevan untuk digunakan. Sehingga bisa dikatakan jika terlalu sedikit fitur hasil *Information Gain* digunakan belum tentu mendapatkan kata yang paling relevan, begitupun dengan terlalu banyak kata digunakan akan memberikan hasil yang tidak optimal.

6.2.3 Analisis Pengaruh Seleksi Fitur

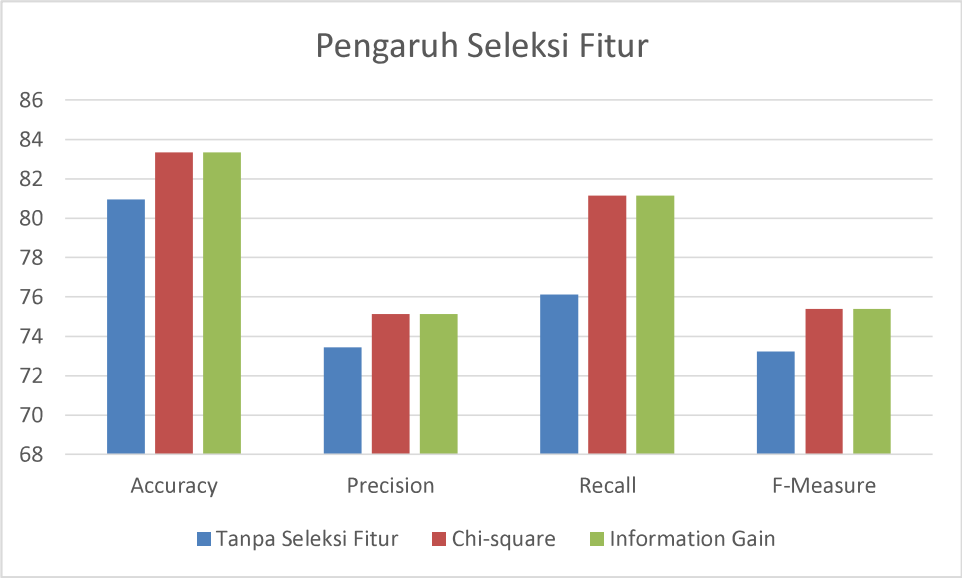
Seleksi fitur berfungsi untuk mengetahui tingkat relevan suatu kata. Dari skenario pengujian menggunakan seleksi fitur *Chi-square* ataupun *Information Gain* didapatkan hasil urutan setiap kata/fitur dari yang paling relevan hingga paling tidak relevean. Pada analisis kali ini akan membandingkan salah satu hasil *Chi-square* yang terbaik yaitu dengan *threshold* 50% dan begitupun dengan *Information Gain* diambil hasil terbaik dengan *threshold* 50%. Untuk hasil urutan 7 fitur tertinggi dan 7 fitur terendah dapat dilihat pada Tabel 6.4.

Tabel 6.4 Hasil Urutan Seleksi Fitur

7 Fitur Tertinnggi *Chi-Square*		**7 Fitur Terendah *Chi-Square***		**7 Fitur Tertinnggi *Information Gain***		**7 Fitur Terendah *Information Gain***	
Term	Nilai *Chi*	Term	Nilai *Chi*	Term	Nilai *IG*	Term	Nilai *IG*
parkir	38,25	rado	1,09	parkir	0,10	martadinata	0,003
karcis	19,32	abadi	1,09	karcis	0,05	madiun	0,003
jalan	15,13	jumlah	1,09	jalan	0,04	persis	0,003
sampah	12,56	akal	1,09	sambat	0,03	ujung	0,003
sambat	12,40	tembok	1,09	liar	0,03	lalan	0,003
liar	11,40	follow	1,09	tukang	0,03	balen	0,003
tukang	9,53	up	1,09	sampah	0,03	kotalama	0,003

Bisa diamati pada Tabel 6.4 jika 7 fitur tertinggi dari nilai *Chi-square* ataupun *Information Gain* merupakan fitur yang relevan untuk digunakan dalam pengaduan SAMBAT *Online* utamanya tiga SKPD yang digunakan pada penelitian kali ini yaitu DIshub, DKP, dan DPUPPB. Kata seperti 'parkir', 'karcis', 'jalan', sampah', 'sambat', 'liar', dan 'tukang' merupakan kata yang dipakai untuk melakukan pengaduan seperti parkir liar, kerusakan jalanan, atau kebersihan lingkungan. Sedangkan fitur yang memiliki nilai *Chi-square* ataupun *Information Gain* yang rendah merupakan kata-kata yang tidak relevan digunakan dalam melakukan pengaduan tertentu atau kata-kata yang sering digunakan pada pengaduan apapun. Jadi, dapat disimpulkan bahwa semakin tinggi nilai suatu fitur/kata, maka semakin relevan juga kata tersebut untuk digunakan. Sebaliknya, semakin rendah nilai suatu fitur/kata, maka semakin tidak relevan juga kata tersebut untuk digunakan.

Setelah mengetahui hasi dari pengujian tanpa menggunakan seleksi fitur dan pengujian menggunakan seleksi fitur, sehingga kita dapat membandingkan dari keduanya untuk mengetahui pengaruh seleksi fitur. Untuk perbandingan hasil tanpa menggunakan seleksi fitur dan menggunakan seleksi fitur dapat dilihat pada Gambar 6.1.



Gambar 6.1 Grafik Pengaruh Seleksi Fitur

Untuk mengetahui pengaruh seleksi fitur terhadap sistem klasifikasi, dapat diamati dari Gambar 6.1. Dimana grafik di atas membandingkan hasil pengujian dari pengujian tanpa menggunakan seleksi fitur dan menggunakan seleksi fitur. Dapat dilihat dari grafik tersebut hasil pengujian dari sisi *accuracy, precision, recall,* maupun *f-measure,* sistem yang melalui proses seleksi fitur baik menggunakan *Chi-square* maupun *Information Gain* menghasilkan hasil yang sama. Hal ini dikarenakan fitur yang diekstraksi sebesar 50% oleh *Chi-Square* dan *Information Gain* merupakan fitur yang sama hanya terdapat perbedaan urutan. Sehingga pada saat dilakukan klasifikasi, fitur yang digunakan sebagian besar sama sehingga tidak mempengaruhi hasil klasifikasinya. Pada grafik juga dapat dilihat, sistem dengan melalui tahapan seleksi fitur hasilnya lebih baik dibanding tanpa menggunakan seleksi fitur. Dengan menggunakan seleksi fitur, didapatkan hasil akurasi sebesar 83,33% dengan ekstraksi fitur masing-masing sebanyak 50%. Seleksi fitur *Chi-Square* maupun *Information Gain* berhasil mengeksraksi fitur yang relevan untuk digunakan sehingga mendapatkan hasil yang lebih baik. Dengan tidak menggunakan seleksi fitur, membuat sistem menggunakan semua fitur yang ada. Hasil yang kurang akurat memungkinkan terjadi karena terdapat beberapa fitur yang muncul di semua kelas (tidak merepresentasikan satu kelas spesifik) sehingga bisa membuat nilai *posterior* atau kelas yang terpilih bukan kelas sebenarnya.

6.3 Skenario Kombinasi Seleksi Fitur

Skenario ketiga adalah mengkombinasikan kedua seleksi fitur yang digunakan yaitu *Chi-Square* dan *Information Gain.* Tujuan dari pengujian ini untuk mengetahui bagaimana pengaruh dari kombinasi seleksi fitur terhadap sitem klasifikasi yang dibangun. Pengujian kombinasi seleksi fitur ini terdapat dua jenis yaitu, menggunakan operasi AND, dan menggunakan operasi OR. Pada pengujian

kali ini menggunakan hasil fitur *Chi-Square* dengan ekstraksi 50% dan *Information Gain* dengan ekstraksi sebesar 50%.

6.3.1 Menggunakan Operasi AND

Pada pengujian kombinasi seleksi fitur yang pertama, digunakan operasi AND untuk mengkombinasikan fitur hasil *Chi-square* dan *Information Gain.* Hasil kombinasi merupakan fitur yang terdapat di dalam kumpulan fitur *Chi-square* dan *Information Gain.* Fitur yang hanya terdapat di salah satu kumpulan fitur saja tidak akan digunakan. Hasil yang diberikan berupa hasil *accuracy, precision, recall,* dan *f-measure.* Untuk hasil dari pengujian dengan menggunakan operasi AND dalam melakukan kombinasi seleksi fitur dapat dilihat pada Tabel 6.5.

Tabel 6.5 Hasil Pengujian Kombinasi Seleksi Fitur Menggunakan Operasi AND

Accuracy	83,33%
Precision	75,12%
Recall	81,14%
F-Measure	75,41%

Tabel di atas merupakan hasil pengujian kombinasi seleksi fitur menggunakan operasi AND. Pada pengujian kali ini mengekstraksi fitur sebanyak 50% dari fitur *Chi-square* dan *Informartion Gain.* Hasil pengujian tersebut dihasilkan nilai *accuracy* yang didapatkan adalah 83,33%, *precision* sebesar 75,12%, *recall* sebesar 81,14%, dan *f-measure* sebesar 75,41%. Selanjutnya akan dilakukan pengujian kombinasi seleksi fitur menggunakan operasi OR.

6.3.2 Menggunakan Operasi OR

Pada pengujian kombinasi seleksi fitur yang kedua, digunakan operasi OR untuk mengkombinasikan fitur hasil *Chi-square* dan *Information Gain.* Hasil kombinasi merupakan fitur yang terdapat di dalam kumpulan fitur *Chi-square* dan *Information Gain.* Selain itu, fitur yang hanya terdapat di salah satu kumpulan fitur saja akan tetap digunakan. Hasil yang diberikan berupa hasil *accuracy, precision, recall,* dan *f-measure.* Untuk hasil dari pengujian dengan menggunakan operasi OR dalam melakukan kombinasi seleksi fitur dapat dilihat pada Tabel 6.6.

Tabel 6.6 Hasil Pengujian Kombinasi Seleksi Fitur Menggunakan Operasi OR

Accuracy	83,33%
Precision	75,12%
Recall	81,14%
F-Measure	75,41%

Tabel di atas merupakan hasil pengujian kombinasi seleksi fitur menggunakan operasi OR. Pada pengujian kali ini mengekstraksi fitur sebanyak 75% dari fitur *Chi-square* dan *Informartion Gain.* Hasil pengujian tersebut dihasilkan nilai *accuracy* yang didapatkan adalah 83,33%, *precision* sebesar 75,12%, *recall* sebesar 81,14%, dan *f-measure* sebesar 75,4%. Hasil pengujian tersebut dapat dilihat hasil sistem menggunakan operasi AND atau OR menghasilkan hasil yang sama atau bisa dikatakan kedua operasi tersebut sama baiknya.

6.3.3 Analisis Pengaruh Kombinasi Seleksi Fitur

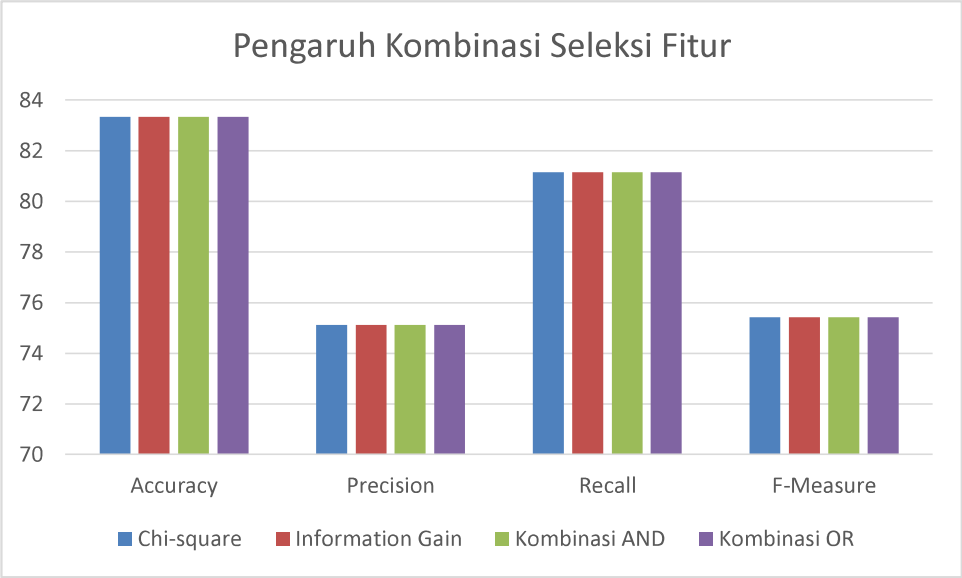
Seperti yang sudah dijelaskan sebelumnya, konsep seleksi fitur bertujuan untuk mengetahui tingkat relevan suatu kata. Kombinasi seleksi fitur diharapkan mampu memadukan dua seleksi fitur untuk mendapatkan kumpulan fitur yang lebih optimal. Hasil kombinasi seleksi fitur menggunakan operasi AND menghasilkan jumlah kumpulan fitur yang lebih sedikit dibanding kumpulan fitur hasil kombinasi menggunakan operasi OR. Perbandingan kumpulan fitur dari kedua operasi dapat dilihat pada Tabel 6.7.

Tabel 6.7 Perbandingan Jumlah Fitur AND dan OR

Jenis Kombinasi	Jumlah Fitur
Operasi AND	760 fitur
Operasi OR	820 fitur

Pada tabel di atas dapat dilihat jumlah fitur yang dihasilkan operasi AND akan lebih sedikit dibanding operasi OR. Terlihat selisih 60 fitur dikarenakan operasi AND hanya mengambil fitur yang terdapat pada kedua seleksi fitur. Pada hasil pengujian, operasi OR sama dengan operasi AND. Hal ini dikarenakan pada fitur AND memuat fitur yang relevan untuk digunakan. Begitupun pada fitur OR walaupun lebih banyak fitur yang digunakan tetapi fitur tersebut yang relevan sehingga hasil pengujiannya sama baiknya.

Dengan didapatkannya hasil pengujian menggunakan kombinasi seleksi fitur, sehingga dapat dilakukan analisa pengaruh kombinasi seleksi fitur pada penelitian kali ini. Perbandingan melakukan kombinasi seleksi fitur dan tanpa melakukan kombinasi, dapat dilihat pada Gambar 6.2.



Gambar 6.2 Grafik Pengaruh Kombinasi Seleksi Fitur

Dari grafik di atas dapat dilihat perbandingan hasil pengujian antara melakukan kombinasi seleksi fitur dan tidak melakukan kombinasi seleksi fitur. Dapat dilihat hasil *accuracy, precision, recall,* dan *f-measure* yang dihasilkan *Chi-square, information Gain,* dan kombinasi menggunakan operasi AND maupun OR sama persis. Fitur dengan hanya menggunakan *Chi-square* atau *Information Gain* akan lebih banyak dikarenakan tidak ada fitur yang dibuang. Sedangkan menggunakan operasi OR menghasilkan lebih banyak lagi fitur karena kedua hasil fitur saling melengkapi. Tetapi pada saat menggunakan operasi AND juga didapatkan hasil yang sama baiknya walaupun hanya menggunakan fitur yang terdapat di kedua hasil seleksi fitur. Sehingga dari hasil pengujian dapat disimpulkan bahwa pengaruh kombinasi seleksi fitur tidak berpengaruh dengan hasil sistem yang diberikan. Hal ini dikarenakan dari sisi *Chi-square* dan *Information Gain* berhasil mengekstraksi fitur yang hampir sama sehingga walaupun dilakukan kombinasi, hasil fitur yang dihasilkan relevan untuk digunakan.

6.4 Skenario Variasi *Threshold* Kombinasi Ekstraksi Fitur

Skenario keempat akan dilakukan variasi *threshold* dalam melakukan ekstraksi fitur. Pengujian ini bertujuan untuk mengetahui berapa *threshold* yang baik digunakan untuk mendapatkan hasil pengujian yang optimal. Fitur yang digunakan merupakan fitur hasil kombinasi dengan operasi AND. Pada pengujian kali ini akan menggunakan 4 macam *threshold* yaitu, 25%, 50%, 75%, dan 100%. Sehingga dapat dibandingkan untuk mendapat *threshold* terbaik.Untuk hasil pengujian variasi nilai *threshold* dapat dilihat pada Tabel 6.8.

Tabel 6.8 Hasil Pengujian Variasi *Threshold*

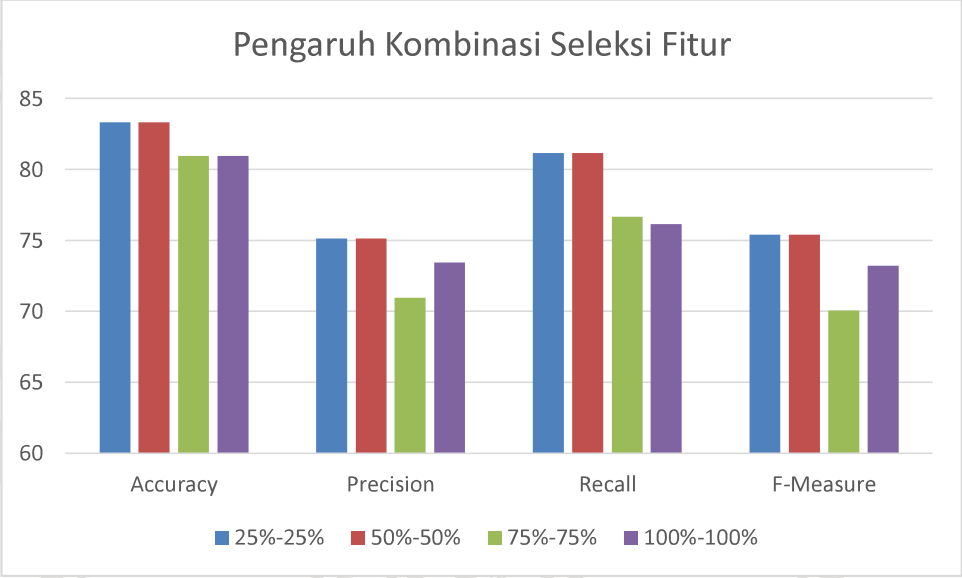
Chi-*Square*	*Information Gain*	*Accuracy*	*Precision*	*Recall*	*F-Measure*
25%	25%	83,33%	75,12%	81,14%	75,41%

50%	50%	83,33%	75,12%	81,14%	75,41%
75%	75%	80,95%	70,95%	76,70%	70,05%
100%	100%	80,30%	73,45%	76,14%	73,23%

Tabel di atas merupakan hasil pengujian menggunakan 4 varian nilai *threshold*. Dari hasil tersebut terlihat pengujian terbaik dihasilkan dengan menggunakan *threshold* sebesar masing-masing 25% dan masing-masing 50%. Kedua nilai *threshold* tersebut memberikan nilai *accuracy* sebesar 83,33%, *precision* sebesar 75,12%, *recall* sebesar 81,14%, dan *f-measure* sebesar 75,14%.

6.4.1 Analisis Pengujian Variasi *Threshold* Kombinasi Ekstraksi Fitur

Setelah melakukan pengujian dengan melakukan variasi nilai *threshold,* didapatkan hasil yang berbeda setiap nilai *threshold*. Untuk perbandingan lebih jelas dari hasil variasi *threshold* pada kombinasi dengan operasi AND dapat dilihat pada Gambar 6.3.



Gambar 6.3 Grafik Variasi Nilai *Threshold*

Pada grafik di atas, dapat diamati bahwa hasil terbaik didapatkan pada nilai *threshold* masing-masing 25% dan masing-masing 50%. Hasil mulai menurun pada saat memasuk *threshold* 75% atau 100%. Hal ini dikarenakan dalam ekstraksi fitur, nilai teratas merupakan yang paling relevan. Sebaliknya pada nilai paling bawah merupakan paling tidak relevan. Sehingga pada saat pemotongan lebih sedikit yaitu 25% dan 50% hanya menggunakan fitur yang paling relevan. Pada fitur 75% dan 100% memungkinkan banyaknya fitur yang tidak relevan sehingga mempengaruhi hasil klasifikasi. Hal ini terjadi karena fitur yang paling relevan merupakan fitur yang hanya merepresentasikan satu kelas tertentu seperti contohnya kata 'parkir' yang sangat merepresentasikan kelas Dishub. Sedangkan kata tidak relevan muncul lebih dari satu kelas seperti contohnya kata 'lapor'. Dengan terpakainya kata yang tidak relevan pada tahapan klasifikasi membuat

hasil *posterior* atau kelas yang terpilih bukan kelas sebenarnya. Sehingga dapat disimpulkan dalam pengujian kali ini didapatkan *threshold* yang paling baik dengan menggunakan masing-masing 25% atau masing-masing 50% dengan alasan yang sudah dijelaskan di atas.

BAB 7 PENUTUP

Bab ini berisi kesimpulan dan saran dari seluruh penelitian yang dilakukan pada klasifikasi teks Bahasa Indonesia pada dokumen pengaduan SAMBAT *online* menggunakan metode *Naïve Bayes* dan kombinasi seleksi fitur, sehingga dapat digunakan untuk pengembangan penelitian selanjutnya.

7.1 Kesimpulan

Setelah dilakukan pengujian dan analisis dari hasil penelitian ini, dapat ditarik beberapa kesimpulan sebagai berikut:

1. Pada saat melakukan klasifikasi menggunakan metode *Naïve Bayes* tanpa melalui proses seleksi fitur, didapatkan hasil akurasi sebesar 80,95%. Sedangkan pada saat dilakukan klasifikasi menggunakan metode *Naïve Bayes* dan sebelumnya dilakukan proses ekstraksi fitur sebanyak 50% menggunakan *Chi-Square,* didapatkan hasil akurasi sebesar 83,33%. Hasil yang sama didapatkan pada saat menggunakan *Information Gain* dalam melakukan ekstraksi fitur sebanyak 50%. Dapat disimpulkan bahwa seleksi fitur berpengaruh terhadap hasil klasifikasi menggunakan metode *Naïve Bayes* untuk mendapatkan yang lebih baik.
2. Pada saat melakukan klasifikasi menggunakan metode *Naïve Bayes* dengan dilakukannya kombinasi seleksi fitur sebelumnya, dimana fitur yang diekstraksi sebesar masing-masing 50% didapatkan hasil akurasi sebesar 83,33% dengan menggunakan operasi AND dalam pengkombinasiannya. Sedangkan menggunakan operasi OR dalam pengkombinasiannya didapatkan hasil akurasi sebesar 83,33%. Sehingga dapat disimpulkan bahwa kombinasi dalam seleksi fitur tidak berpengaruh untuk menghasilkan hasil yang lebih baik dikarenakan hasil yang diberikan sama dengan hasil apabila tidak dilakukan kombinasi.

7.2 Saran

Untuk pengembangan selanjutnya, terdapat beberapa saran dari penulis sebagai berikut:

1. Untuk pengembangan selanjutnya dapat menggunakan metode seleksi fitur yang lainnya untuk mengetahui apakah metode seleksi fitur yang lainnya dapat memberikan hasil yang lebih baik atau tidak.
2. Untuk pengembangan selanjutnya dapat menggunakan metode klasifikasi yang lainnya untuk mengetahui apakah metode klasifikasi yang lainnya dapat memberikan hasil yang lebih baik atau tidak.
3. Kombinasi dengan menggunakan operasi AND ataupun OR belum mampu memberikan hasil yang lebih baik, sehingga disarankan untuk melakukan kombinasi dengan operasi yang berbeda dengan harapan mendapatkan hasil yang lebih baik.

DAFTAR PUSTAKA

Destuardi & Sumpeno, S., 2009. Klasifikasi Emosi Untuk Teks Bahasa Indonesia Menggunakan Metode *Naïve Bayes,* [online] Tersedia di : <https://reasearchgate.net/> [Diakses 26 November 2018].

Fauzi, M. A., Arifin, A. Z., Gosaria, S. C. & Prabowo, I. S., 2017. *Indonesian News Classification Using Naïve Bayes and Two-Phase Feature Selection Model*. *Indonesian Journal of Electrical Engineering and Computer Science*, [e-journal] 8(3), pp. 610-615. Tersedia melalui: <http://www.iaescore.com/> [Diakses 13 Agustus 2018].

Hamzah, A., 2012. Klasifikasi Teks Dengan Naïve Bayes Classifier (NBC) Untuk Pengelompokan Teks Berita dan Abstrak Akademis, [online] Tersedia di: < http://repository.akprind.ac.id/sites/files/conference-proceedings/2012/hamzah_15430.pdf >

Hidayatullah, A. F. & Ma'arif, M. R., 2016. Penerapan *Text Mining* dalam Klasifikasi Judul Skripsi. Seminar Nasional Aplikasi Teknologi Informasi (STANi). Yogyakarta, Indonesia, 6 Agustus 2016.

Indriati & Ridok, A., 2016. *Sentiment Analysis for Review Mobile Applications Using Neighbor Method Weighted & K-Nearest Neighbor (NW-KNN)*. *Journal of Enviromental Engineering and Sustainable Technology*, [e-journal] 3(1), pp. 23-32. Tersedia melalui: <http://jeest.ub.ac.id/> [Diakses 25 Agustus 2018].

Khadim, A. I., Chea, Y. & Ahamed, N. H., 2014. *Text Document Preprocessing and Dimension Reduction Techniques for Text Document Clustering. 4th International Conference on Artificial Intelligence with Application in Engineering and Technology.* 2014.

Li, Z., Shang, W. & Yan, M., 2016. *News Text Classification Model Based on Topic Model,* [online] Tersedia di : <https://ieeexplore.ieee.org/> [Diakses 20 Agustus 2018].

Manning, C, Raghavan, P, dan Schutze, H. 2009. *Introduction to Information Retrieval.* Cambridge: Cambridge University Press. [pdf]. Tersedia di: <https://nlp.stanford.edu/IR-book/pdf/13bayes.pdf> [Diakses 05 Oktober 2018]

Nugroho, D. G., Chrisnanto, Y. H. & Wahana, A. 2016. Analisis Sentimen Pada Jasa Ojek *Online* Menggunakan Metode *Naïve Bayes,* [online] Tersedia di : <https://publikasiilmiah.unwahas.ac.id/index.php/PROSIDING_SNST_FT/article/view/1526> [Diakses 16 Juli 2018]

Prasanti, A. A., Fauzi, M. A. & Furqon, M. T., 2017. Klasifikasi Teks Bahasa Indonesia Pada Dokumen Pengaduan SAMBAT Online Menggunakan Metode N-Gram Dan Neighbor Weighted K-Nearest Neighbor (NW-KNN). Malang: Universitas Brawijaya.

Pratiwi, F. I., Mardi, R. R. & Setiawan, B. D., 2017. Klasifikasi Topik Pada Skripsi Berdasarkan Judul Dan Abstraksi Dengan Menggunakan Metode *Transformed Weight-Normalized Complement Naïve Bayes*. Malang: Universitas Brawijaya.

Putra, I. B. G. W., Sudarma, M. & Kumara, I. N. S., 2016. Klasifikasi Teks Bahasa Bali Dengan Metode *Supervised Learning Naive Bayes Classifier,* [online] Tersedia di : <https://text-id.123dok.com/> [Diakses 17 September 2018]

Saeys, Y., Abeel, T. & Peer, Y. Vd. 2008. *Robust Feature Selection Using Ensemble Feature Selection Techniques. Indonesian Journal of Electrical Engineering and Computer Science,* [e-journal] pp. 313-325. Tersedia melalui: <http://www.iaescore.com/> [Diakses 17 Semptember 2018].

Somantri, O. & Hasta, I. D., 2017. Implementasi *e-Government* Pada Kelurahan Pesurungan Lor Kota Tegal Berbasis Service Oriented Architecture, [online] Tersedia di: < http://ejournal.poltektegal.ac.id/ > [Diakses 4 April 2018].

Suharno, C. F., Fauzi, M. A. & Perdana, R. S., 2017. Klasifikasi Teks Bahasa Indonesia Pada Dokumen Pengaduan SAMBAT Online Menggunakan Metode K-Nearest Neighbor Dan Chi-Square. Malang: Universitas Brawijaya.

Sun, Changqiu, Xiaolong Wang, dan Jun Xu. 2009. *Study on Feature Selection in Finance Text Categorization. Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics*. San Antonio.

Zheng, Z., Wu, X. & Srihari, R., 2014. *Feature Selection for Text Categorization on Imbalance Data. ACM SIGKKD Exploration Newsletter – Special Issue on learning from imbalance datasets*, [online] Tersedia di: <https://dl.acm.org/citation.cfm?id=1007741> [Diakses 29 Juni 2018].